



**Calhoun: The NPS Institutional Archive**  
**DSpace Repository**

---

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

---

1986

Image interpretation using an expert system.

Rueda, Diego L.

---

<http://hdl.handle.net/10945/21899>

---

*Downloaded from NPS Archive: Calhoun*



Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

**Dudley Knox Library / Naval Postgraduate School**  
**411 Dyer Road / 1 University Circle**  
**Monterey, California USA 93943**

<http://www.nps.edu/library>











# NAVAL POSTGRADUATE SCHOOL

Monterey, California



## THESIS

IMAGE INTERPRETATION USING AN EXPERT  
SYSTEM

by

Diego L. Rueda

December 1986

Thesis Advisor

Chin-Hwa Lee

Approved for public release; distribution is unlimited.

T232241



## REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY			3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE				
4 PERFORMING ORGANIZATION REPORT NUMBER(S)			5 MONITORING ORGANIZATION REPORT NUMBER(S)	
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School		6b OFFICE SYMBOL (if applicable) 590	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000			7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	
8a NAME OF FUNDING/SPONSORING ORGANIZATION		8b OFFICE SYMBOL (if applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c ADDRESS (City, State, and ZIP Code)			10 SOURCE OF FUNDING NUMBERS	
			PROGRAM ELEMENT NO	PROJECT NO
			TASK NO	WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) IMAGE INTERPRETATION USING AN EXPERT SYSTEM				
12 PERSONAL AUTHOR(S) Lt. Rueda, Diego L.				
13a TYPE OF REPORT Master's Thesis		13b TIME COVERED FROM _____ TO _____	14 DATE OF REPORT (Year, Month, Day) 1986 December	15 PAGE COUNT 102
16 SUPPLEMENTARY NOTATION				
17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP	Expert System, Photogrammetry, Aerial Image interpretation.	
19 ABSTRACT (Continue on reverse if necessary and identify by block number) The purpose of this thesis is to study and enhance an automatic image feature analysis system for aerial photographs in order to detect changes with respect to a model defined in a knowledge-base. The system interpretation must resemble some intelligent human interpretation and by its use should be able to reduce the manual effort in processing large volumes of data. This computerized rule-based system is integrated by using Prolog and Image processing operators, which run in a multiprocess environment and communicate through a blackboard storage. The Prolog expert provides the model through a collection of certainty rules and facts, and uses its "inference engine" capability to combine different measurements. They are obtained as the evidence necessary to deduce a conclusion about the condition of a feature.				
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS			21 ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL Chin-Hwa Lee			22b TELEPHONE (Include Area Code) 408-646-2190	22c OFFICE SYMBOL 590



The system must be able to deal with uncertainties generated by noise in the images, variability of imaging conditions, and possible errors in the model. The main task is the formulation of interpretation rules so that the expert system can mimic the "reasoning" effectively from the domain principles and yield high confidence results.

A system that satisfies some of the stated specifications has already been developed by Prof. Chin-Hwa Lee and Hsi-Jian Lee. This thesis as the continuation of the investigation process addresses the following objectives:

1. Analyze in-depth all the Fortran programs in order to detect possible errors in the Image Processing operators (The main modules are: Shadow Processing, Model guided Split and Merge, Region Selection and Model Projection); additionally the documentation of all the programs should be generated.
2. Expand the Rule Data base and Facts Data base to enhance the actual capabilities of the expert interpretation system.
3. Measure the CPU time for each of the subprocesses in order to determine the time consuming parts of the system. New approaches to improve those parts should be considered and developed.
4. Apply the system to different kinds of images to evaluate the system response to several illumination conditions and different image detail.
5. Develop and Incorporate reflectance modelling capabilities to make the system more accurate.

Approved for public release; distribution is unlimited.

Image Interpretation using an Expert System

by

Diego L. Rueda  
Lt, Ecuadorean Navy  
B.S., Escuela Superior Naval, Ecuador, 1976

Submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL  
December 1986

---

## ABSTRACT

The purpose of this thesis is to study and enhance an automatic image feature analysis system for aerial photographs in order to detect changes with respect to a model defined in a knowledge-base. The system interpretation must resemble some intelligent human interpretation and by its use should be able to reduce the manual effort in processing large volumes of data.

This computerized rule-based system is integrated by using Prolog and Image processing operators, which run in a multiprocess environment and communicate through a blackboard storage. The Prolog expert provides the model through a collection of certainty rules and facts, and uses its "inference engine" capability to combine different measurements. They are obtained as the evidence necessary to deduce a conclusion about the condition of a feature.

The system must be able to deal with uncertainties generated by noise in the images, variability of imaging conditions, and possible errors in the model. The main task is the formulation of interpretation rules so that the expert system can mimic the "reasoning" effectively from the domain principles and yield high confidence results.

A system that satisfies some of the stated specifications has already been developed by Prof. Chin-Hwa Lee and Hsi-Jian Lee. This thesis as the continuation of the investigation process addresses the following objectives:

1. Analyze in-depth all the Fortran programs in order to detect possible errors in the Image Processing operators (The main modules are: Shadow Processing, Model guided Split and Merge, Region Selection and Model Projection); additionally the documentation of all the programs should be generated.
2. Expand the Rule Data base and Facts Data base to enhance the actual capabilities of the expert interpretation system.
3. Measure the CPU time for each of the subprocesses in order to determine the time consuming parts of the system. New approaches to improve those parts should be considered and developed.
4. Apply the system to different kinds of images to evaluate the system response to several illumination conditions and different image detail.
5. Develop and Incorporate reflectance modelling capabilities to make the system more accurate.

## THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

## TABLE OF CONTENTS

I.	INTRODUCTION .....	12
A.	THE BACKGROUND .....	12
B.	EXPERT SYSTEM IMPLEMENTATION .....	13
II.	SYSTEM GENERATION .....	15
A.	THE APPROACH TO SOLVE THE PROBLEM .....	15
B.	HOW THE CHANGE DETECTION SYSTEM WORKS .....	18
III.	PHOTOGRAMMETRY TOPICS .....	20
A.	BASIC GEOMETRY OF THE PHOTOGRAPH.- CONCEPTS .....	20
IV.	THESIS DEVELOPMENT .....	25
A.	INTRODUCTION .....	25
B.	SOFTWARE REVIEW .....	25
C.	MEASURING THE CPU TIME .....	28
D.	INCORPORATING THE REFLECTANCE MODELLING .....	30
V.	CONCLUSIONS .....	34
APPENDIX A: PROLOG ENVIRONMENT SUMMARY .....		35
1.	RULE BASE EXPERT .....	35
2.	FACTS DATABASE .....	36
3.	CONTROL RULE #1 .....	37
4.	CONTROL RULE #2 .....	37
5.	CONTROL RULE #3 .....	38
6.	CONTROL RULE #4 .....	39
7.	CONTROL RULE #5 .....	39
8.	CONTROL RULE #8 .....	40
9.	CONTROL RULE #9 .....	41
10.	CONTROL RULE #10 .....	41



11.	CONTROL RULE #11 .....	42
12.	CERTAINTY OF TONE .....	43
13.	CERTAINTY OF SIZE .....	43
14.	CERTAINTY OF SHADOW .....	44
15.	CERTAINTY OF POLYGON .....	45
16.	PROLOG COMMON ROUTINES .....	46
APPENDIX B: COMPUTATIONAL ENVIRONMENT SUMMARY. ....		47
1.	PROGRAM (FORTRAN) 'SELREG' .....	47
2.	PROGRAM (FORTRAN) 'MGSM' .....	50
3.	PROGRAM (FORTRAN) 'SM' .....	52
4.	PROGRAM (FORTRAN) 'SHDPRO' .....	54
5.	PROGRAM (FORTRAN) 'CRBKBD' .....	58
6.	PROGRAM (FORTRAN) 'PTPROJ' .....	62
7.	SUBROUTINE (FORTRAN) 'AFFIJ' .....	64
8.	SUBROUTINE (FORTRAN) 'JCPUT' .....	65
9.	SUBROUTINE (FORTRAN) 'PROJMBR' .....	66
10.	SUBROUTINE (FORTRAN) 'REFINE' .....	67
11.	SUBROUTINE (FORTRAN) 'BLKBRD' .....	71
12.	SUBROUTINE (FORTRAN) 'PROJECT' .....	73
13.	SUBROUTINE (FORTRAN) 'XY2IJ' .....	75
14.	SUBROUTINE (FORTRAN) 'CONVERT' .....	77
15.	SUBROUTINE (FORTRAN) 'SHADOW' .....	79
16.	SUBROUTINE (FORTRAN) 'FILENAM' .....	80
17.	SUBROUTINE (FORTRAN) 'QUADEQ' .....	81
18.	SUBROUTINE (FORTRAN) 'MEASREG' .....	82
19.	SUBROUTINE (FORTRAN) 'TONE' .....	84
20.	SUBROUTINE (FORTRAN) 'WRPROP' .....	85
21.	SUBROUTINE (FORTRAN) 'MEASONE' .....	86
22.	SUBROUTINE (FORTRAN) 'NEIGHREGS' .....	88
23.	SUBROUTINE (FORTRAN) 'ASSREG' .....	89
24.	SUBROUTINE (FORTRAN) 'REFLEC' .....	90
25.	FUNCTION (FORTRAN) 'PLUMB' .....	93
26.	SUBROUTINE (PASCAL) 'REGFILL' .....	94

APPENDIX C: PROGRAM REFLEC .....	96
LIST OF REFERENCES .....	99
INITIAL DISTRIBUTION LIST .....	101

## LIST OF TABLES

1. MODEL STRUCTURE .....	16
2. SOFTWARE STRUCTURE .....	26
3. CPU TIME CONSUMPTION (IN SECONDS) .....	29

## LIST OF FIGURES

3.1	Image and object spaces . . . . .	21
3.2	Image and object coordinate systems . . . . .	23
4.1	Local geometry of the incident and reflected rays . . . . .	32

## ACKNOWLEDGEMENTS

I would like to express my gratitude to my Thesis Advisor, Dr. Chin-Hwa Lee, for his advice and assistance in completion of this Thesis.

Also I would like to thank Dr. Charles W. Therrien and others that directly or indirectly contributed for the accomplishment of this Thesis.

Finally I am grateful to my parents Pablo and Maria, and specially to my wife Maria del Rosario and my son Diego Andres, whose love and affection has been the force that helped me to achieve this goal of education.



## I. INTRODUCTION

### A. THE BACKGROUND

To have automatic image feature analysis capabilities over an aerial photograph, involves locating and studying specific objects. In this analysis, locating buildings and detecting changes in their structure is the objective. These are difficult tasks because of the variability in image conditions such as weather, time and attitude when the photograph was taken. Other possible reasons are the introduction of noise in the digitized image, errors in the 'model' used to match the observations, and errors in the results generated by the Change Detection process itself.

There have been several approaches to generate model-based image interpretation systems, see (Refs. 1,2,3). These efforts have tended to concentrate on high-level issues such as three dimensional object representation and matching. Others have been involved in aerial photograph analysis, see (Refs. 4,5,6). The need to have automatic image interpretation systems is related directly to the large number of images obtained from satellites and intelligence sources. Updating the information provided by these photographs requires the employment of human resources and time to analyze, detect and identify changes in existing buildings or to establish the creation of new ones. Because this process doesn't guarantee its results to be free of errors, it is time consuming and takes large amounts of human resources. Consequently the use of an automatic Image Interpretation System is justified.

If an Expert System could be created to perform automatically the image feature interpretation, the same process could also be used to identify other specified features. Its principles could be applied to a wider number of applications like the development of artificial vision systems. The system developed by Dr. Chin-Hwa Lee and Hsi-Jian Lee actually satisfy the main objectives of this investigation. The thesis effort should be oriented to continue the implementation of new capabilities and to make this system more efficient in terms of execution speed and memory usage. To obtain these objectives, is necessary to review the software in detail and obtain a better understanding of how the system works in order to point out the possible areas of enhancement. The work to be done here will be considered as the necessary steps to a better and more sophisticated method of image interpretation.

The contents of this thesis include:

Chapter I: A brief description of the problem and advantages of using expert systems for the implementation of the Image Interpretation.

Chapter II : A description of how an expert system could be created to perform the required objectives, also has a general overview of the actual system structure.

Chapter III : Some concepts on photogrammetry. This background is required for studying the system, because the analysis is done on aerial photographs.

Chapter IV : Shows the work done through the development of this thesis and includes details on the execution of predetermined tasks.

Chapter V : Gives the conclusions obtained after working with the Image Interpretation System.

## **B. EXPERT SYSTEM IMPLEMENTATION**

Expert systems are computer programs that emulate the behavior of human experts in a specified domain of knowledge, they use knowledge and inference procedures to solve problems that are difficult enough and require significant human expertise for their solutions, see (Ref. 7).

Some capabilities found in the expert systems are needed to develop our image interpretation system as the followings: they are 'evolvable', that is, easy to modify and update, they also follow a top-down modular design approach where modules are used to hide details and through their use later changes in the design are easy to make thus keeping the knowledge base complete and accurate. They have the capability of both forward and backward reasoning. Forward chaining is a problem solving paradigm in which chaining starts from a set of conditions and moves toward some conclusion. Backward chaining is a process that requires a goal or a hypothesis as a starting point, from where the process proceed 'working backwards' along a path to determine if the hypothesis is true. Improvement in the performance of the program is possible by constraining the search for a solution. Additionally expert systems allow uncertainties to be taken into account during the design of the system.

The structure of the Image Interpretation system consists of knowledge clauses: rules, facts and structures; a reasoning element: deductive inference engine and control programs; and image processing routines: to process the image and work as the computational environment.

The knowledge base approach allows us to use the expert knowledge of the domain by developing change detection rules and incorporating a-priori knowledge, see (Ref. 8). The Rule Base is an open ended structure which permits information to be added or subtracted in an incremental manner. These are tasks performed as a consequence of finding mistakes in the process and improving the organization of the rules, see (Ref. 9). These changes must be done in the contents of the control structure in an efficient manner, therefore the system must have a modular implementation to facilitate evolutionary changes. The main task is to provide the system with the knowledge of expertise in the field of image interpretation, then the computer makes use of this knowledge, as in this system, to detect changes in buildings located in aerial photographs.

Expert systems software show greater separation between data and program constructs. This feature is generated by providing non-procedural constructs at higher levels. This separation enables more flexibility in the creation and maintenance of software as well as ease of understanding, see (Ref. 10). An expert system also has the following three characteristics that makes it convenient for our application. First, an expert has knowledge of the concepts relevant to the domain and can solve problems by reasoning from the domain principles. It is able to capture and retain expertise from an expert in the field, which will be used in the interpretation of the results. Second, because knowledge-based systems attempt to represent expertise explicitly, the system makes decisions, or suggest decisions, and explain its conclusions. This explicit representation of symbolic structures, behavior and reasoning allows flexibility and adaptability as a result of the explicit declarative nature of the representation. Third, the knowledge representation and use are embeded in the program's control flow. In contrast to traditional procedural programming languages, such as Fortran and Pascal, that have been used previously to implement similar systems, these systems, even being efficient, lack transparency, modularity and flexibility. Consequently adding, deleting or updating the knowledge is a difficult task.

## II. SYSTEM GENERATION

### A. THE APPROACH TO SOLVE THE PROBLEM

Analysis of aerial photographs is a complex task. A prime cause is the presence of fine texture which causes difficulties for the low level processes such as the segmentation routines, see (Refs. 11,4). Another source of difficulty is that the desired objects and structures may be too small compared to the resolution of the photograph see (Refs. 12,13). A detailed analysis of a complete high resolution aerial image is generally prohibitive because of the computational costs.

In order to overcome the difficulties in the brute force analysis of aerial photographs and obtain a better description of the area under consideration, we are using artificial intelligence techniques. The system must emphasize a goal-directed approach and reasoning techniques with possibilities of using a-priori knowledge from the 'model' and data calculated from the image to achieve the determined goals.

The approach to solve the change detection problem, demands the establishment of a 'model' of the object, which will be used as the guide element, a Rule Base expert that has control structure for the Change Detection System, and Image Processing operators that provide the necessary measurements required to achieve goals of the Prolog Interpreter.

The term 'model' will be used to describe the building or group of buildings that we are trying to verify in the image. The facts database used as the environmental 'model' is a tree like structure composed of a number of nodes, where each node is described uniquely, see (Table 1). The environmental 'model' has a hierarchical structure where a-priori information describes in a set of facts about their individual properties and interrelationships. Besides the facts database, there must be several dynamic databases oriented to follow the development of the process and to record the intermediate changes being performed.

Every node is described as a collection of related properties. It includes appearance knowledge in the form of object properties : predicted location, surface material category, children and ancestor nodes, minimum boundary rectangular, background SMC, and properties to be analyzed in this node: shadow, size and tone, and weights assigned to these properties. The height, number of vertices, and

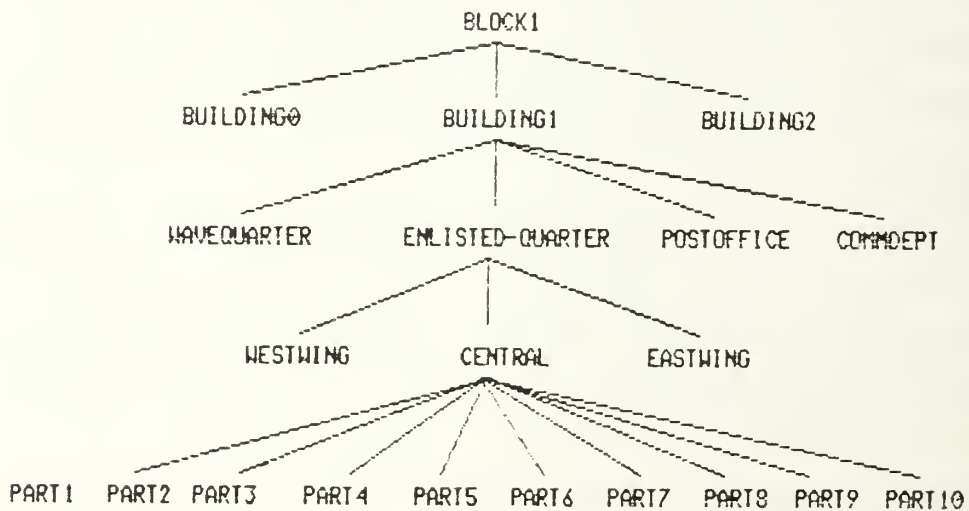


TABLE 1  
MODEL STRUCTURE

## HIERARCHY LEVELS

1. Block
2. Building cluster
3. Building
4. Building parts
5. Subparts

## REPRESENTATION OF THE 'MODEL'





compactness must also be included in the node description. The facts database can specify different weights for the assigned properties in the nodes. Some properties are less reliable, conversely, other properties are considered stable and therefore given more weight.

The control structure, in this case the Rule Based expert exhibit a backward chaining of inference as it tries to achieve a goal. The importance of this process relies on the rules not calling to each other. They communicate through the facts in the path to satisfy a goal, and make use of newly generated assertions. The Rule Based expert has relational knowledge in the form of parameterized relations between objects. It can adapt the more general 'model' to any particular image by a reasoning like process. This resemble an intelligent human interpretation process.

The system must use its rules to combine the region description attributes and select thresholds to modify its own results according to the image conditions. In case of failure, the set of rules must reason about the situation, tailor its behavior to redefine the analysis scheme, and obtain better results. The system must be able to deal with uncertainties that may originate from the imprecise 'model' or from the noise data generated by the image processing operators.

The Image Processing operator also provides the knowledge base with measured relationships obtained from the segmented image. They are: the labeled region map, number of regions detected in the image, sun direction, base plane equation coefficients, area, perimeter, average tone and tone variance, minimum boundary rectangle and compactness. Also included are general information of the image as 'affine' coefficients, exposure station, sun direction and the indicator to the corresponding side of the stereo pair of images being analyzed.

The image processing operators include standard image processing routines written in Fortran or Pascal languages. They consist of region segmentation algorithms, shadow processing, and region analysis capabilities. The recognition of objects are done in several processes which are designed to detect the specific type of objects in a prespecified region. In this system, the image description must be measured automatically in the computational environment and is particular to the photograph being analyzed. The a-priori knowledge provides the Change Detection System with predicted area facts, that helps to locate the area of the buildings.

The Change Detection System studies the correspondence between the image and the 'model' of the domain of interest, which in this case describe the knowledge about

buildings. Once the 'model' has been consulted by the Prolog Interpreter, the process can be then divided in two main parts: the first is the segmentation of the image into regions, where their properties are stored into the background interprocess communication area shared by all subroutines of the computational environment; the second part is the matching and inference process.

A feature being analyzed may be broken into numerous region segments, see (Ref. 14) not necessarily matching the predicted model projections. Therefore, the segmentation will be crucial to the success of the system behavior and additional efforts will be required to refine the segmentation of the image. Here, the segmentation process is knowledge directed and this could decide the success or failure of the analysis.

To summarize, in the Change Detection System there are three levels of interaction. The first being the 'model' is the knowledge base that provides the system with the building description, the search can be constrained according to its predicates. The second level, being the Rule Base Process is the one that decides which process to initiate to satisfy system goals. It is the reasoning part that tries to find the regions with the desired characteristics. The Rule Base process can also react to failure by readapting itself and trying to resatisfy its goals. Finally, the third level, which consists of image processing operators, examine features and performs computations on the image itself.

These three elements interact and generate the evidence necessary to deduce a conclusion reflecting the situation of a feature. After evaluation, if the calculated results are different from the expected, the system reevaluates its conditions and takes corrective action.

## **B. HOW THE CHANGE DETECTION SYSTEM WORKS**

After the facts database has been generated for the available buildings and the Rule Base expert established in the Change Detection System, the digitized aerial photograph is segmented into regions using a segmentation algorithm. The segmentation algorithm is provided by the Spider library; a software library that contains image processing operators. The characteristics of every region to be analyzed are extracted. Now, by using the building's predicted area location obtained from the facts database, its position is verified in the digitized image in order to decide whether to continue with the image interpretation analysis. The 'blackboard' interprocess

communication area is created, and the extracted region characteristics are also read by the Change Detection System to be included in the facts database.

Now the Change Detection System generates a list of nodes as subjects of interest to establish the existence of the buildings and to obtain the partial evidences necessary to deduce a general conclusion. The nodes in the list (referenced as 'checkinglist') will be activated progressively from the lowest to the upper hierarchy in the tree structure.

The certainty value of every node is calculated by combining the weighted properties of the node and multiplying it with the arithmetic average of the certainty value from all the children nodes. If this is a terminal node, its father's certainty should be updated and its descendants removed from the 'checkinglist'. This process continues until all nodes have been verified. Once the certainty value of a node is decided, its status is reported. If the results do not yield enough confidence for the existence of a specified node, then the segmented image is refined by a 'model' guided split and merge process. The split and merge process uses the corresponding region associated with the node projection. It implies the creation of new regions related to the segmented areas generated by overlapping the node projection over the labeled region map. After the refining process, the 'blackboard' and the facts database are updated with the new region characteristics.

The Prolog expert then repeat the process of obtaining the certainty values of the node according to tone, size, shape and shadow characteristics in the following ways:

*The certainty of size* is obtained according to the relation of node projection area against the sum of the corresponding region's area.

*The certainty of polygon* uses the mapped area of the node against the sum of the intersecting segments in the labeled region map.

*The certainty of shadow* is obtained by first obtaining the location in which to look for the shadow. This process uses implicit knowledge of the sun's position as provided in the common area BOARD, it then uses the image processing operators to obtain the tones and areas of the predicted projection shadow areas, and evaluates them against the ones measured from the digitized image.

*The certainty of tone* also makes use of the sun's position and image conditions already known to calculate the tone of the regions of different nodes. It uses the surface material category and surface normal to determine the amount of illumination reflected from the node.

### III. PHOTOGRAMMETRY TOPICS

The following definitions and concepts are included because they are vital to the understanding of the Change Detection System. Many of the parameters discussed here are used in different programs and subroutines in the Change Detection System. These concepts have been summarized from a text in photogrammetry, see (Ref. 15).

#### A. BASIC GEOMETRY OF THE PHOTOGRAPH.- CONCEPTS

##### 1. Orthographic versus Perspective Projection

A map is a reproduction at reduced scale of an orthographic projection from the terrain onto a reference datum plane. A *perspective* or *central projection*, is one in which all points are projected onto the reference plane through one point called the *perspective center*. In photography the sensitized film occupies the reference plane where the negative is formed behind the perspective center, see (Fig. 3.1).

##### 2. Image and Object Spaces

The photograph is a perspective projection, and the optical center of the camera lens is the perspective center. The region between the perspective center and the negative of the photograph is called the *image space*. The region occupied by the object to be photographed, including the reference coordinate system, is called the *object space*.

##### 3. Image Coordinates

The plane of the photograph, which contains images of object points, is a 2-D representation of a generally 3-D object space. The X and Y axes of the photograph are defined by the fiducial marks, with the center of collimation called the *indicated principal point IPP* or *fiducial center* taken as the origin. The position of the perspective center will be over the IPP at a focal distance 'f'.

##### 4. Object Coordinates

A 3-D right handed Cartesian Coordinate System X, Y, Z is used as reference for the object space. If 'L' is called the perspective center, then the object coordinates of the perspective center, or better said the exposure station is XL, YL, ZL as described in the PROJECT and XY2IJ subroutines see Appendix B and (Ref. 16).



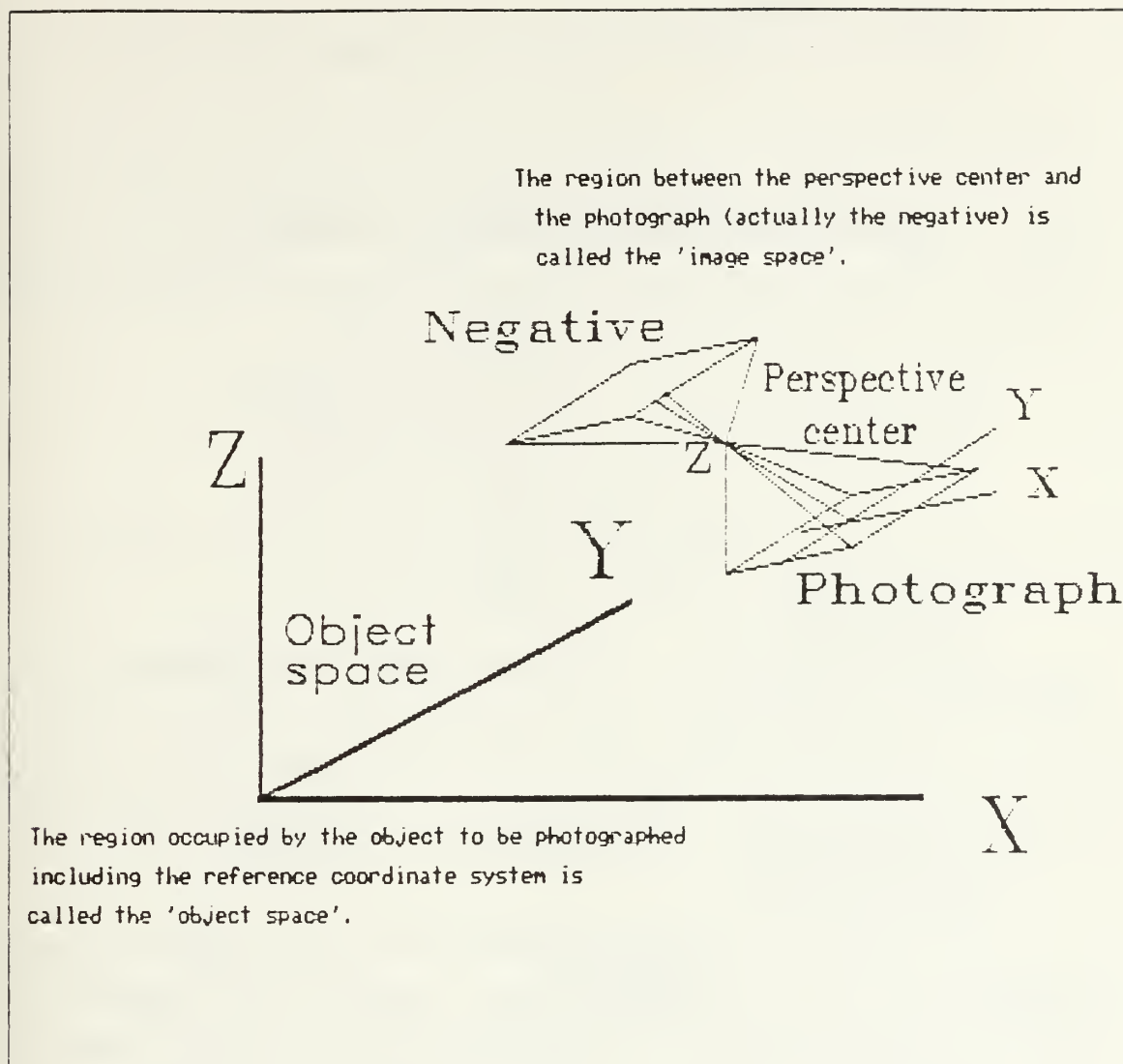


Figure 3.1 Image and object spaces.

## 5. Interior Orientation

The interior orientation of a photograph defines the form of the bundle of rays emerging from the perspective center to the points in the 'image' space. It expresses the angular relationship between object space rays on the basis of the location of the image points with respect to the perspective center. Thus the position of the perspective center with respect to the image coordinate system represents the geometric elements of interior orientation, being  $X_0$ ,  $Y_0$  and  $F$ , where  $X_0$  and  $Y_0$  are the 2-D coordinates of the IPP.



## 6. Exterior Orientation

The exterior orientation of the photograph defines the *position and orientation* in the object space. The position of the photograph is defined by the object space coordinates of the perspective center XL, YL, ZL. The orientation which describes the attitude of the camera at the moment of exposure, refers to the spatial relationship between the object coordinate system and the image coordinate system. The system used here, and a very common system, defines the orientation by the *orientation angles* OMEGA, PHI, KAPPA as used in the PROJECT subroutine.

OMEGA is the first rotation around X, counterclockwise.

PHI is the second rotation around Y, counterclockwise.

KAPPA is the third rotation around Z, counterclockwise.

Thus the exterior orientation of the photograph is defined in terms of XL, YL, ZL, OMEGA, PHI, and KAPPA.

## 7. Orientation Matrix

The relationship between the image and the object coordinate system is expressed by a 3 x 3 orthogonal matrix where each one of its elements is a function of either one of the angles OMEGA, PHI, KAPPA. This matrix allows to transform from the object space system to the image space system, that is from the real world to the photograph, see (Fig. 3.2).

## 8. Collinearity Equations

Any individual pixel in the photograph can be mapped from a 3-D object coordinates by use of the 3 elements of interior orientation and 6 elements of the exterior orientation, which describe the collinearity equations. These 9 elements are constant for all pixel in the image, but since the photograph is a 2-D representation of a 3-D object, the scale is different from one pixel to another. To avoid this problem, we use the collinearity equations as shown in the subroutine PROJECT.

## 9. Photo to Digital Image Transformation

Transformation from the photograph coordinates to the digital image coordinates is called an 'affine' transformation and includes a rotation, two scale changes and two translations. This implies the solution of the equations:

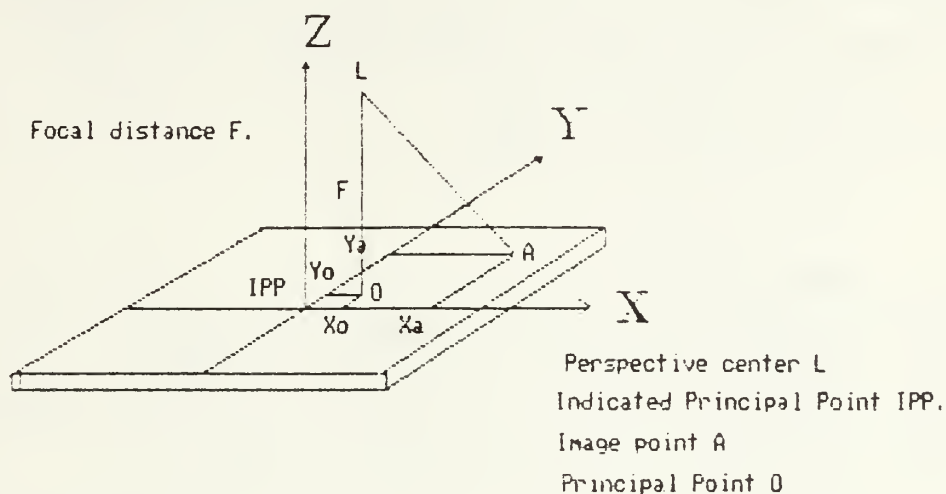
$$XP = SX(\cos(\beta) - \delta \sin(\beta))XI - (SY \sin(\beta))YI + C1$$

$$YP = SX(\sin(\beta) + \delta \cos(\beta))XI + (SY \cos(\beta))YI + C2$$

where

$XP, YP$  = Photo coordinates

## IMAGE COORDINATE SYSTEM



## OBJECT COORDINATE SYSTEM

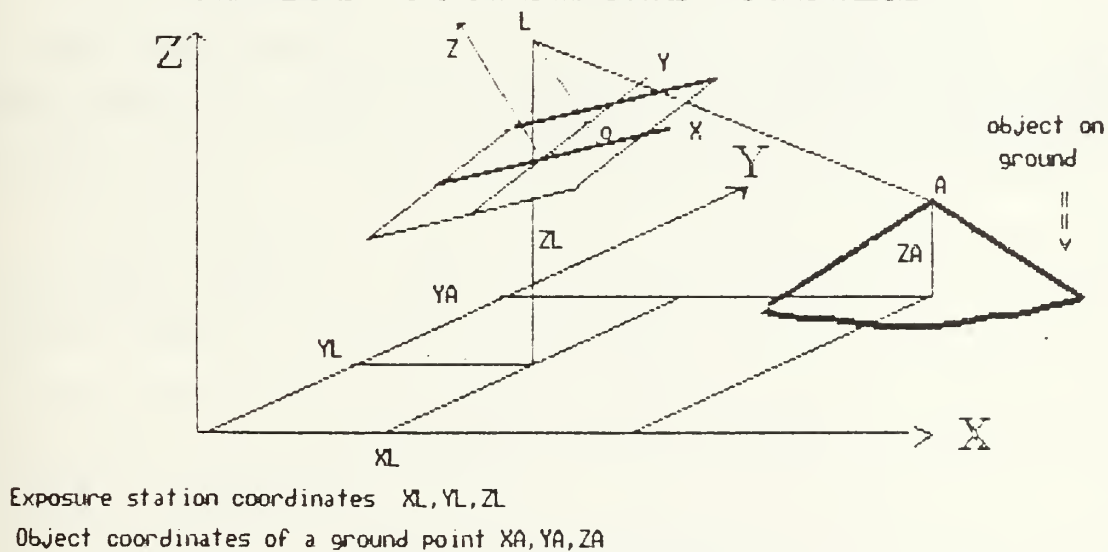


Figure 3.2 Image and object coordinate systems.

$XI, YI$  = Image coordinates.

$SX, SY$  = Scale changes.

$C1, C2$  = Translation in both axes.

$DELTA$  = Nonperpendicularity or affinity between the axes  $X, Y$ .

$BETA$  = Rotation angle.

This transformation is also called 6-parameter transformation. This process has been implemented in the subroutine XY2IJ. The following data has been provided for the analysis of the aerial photographs under test:

$$A1 = SX(\cos(\beta) - \delta \sin(\beta))$$

$$A2 = SY(\sin(\beta) + \delta \cos(\beta))$$

$$B1 = -SY\sin(\beta)$$

$$B2 = +SY\cos(\beta)$$

#### 10. Data provided

- (a) Calibrated focal length for both photos in millimeters.
- (b) Calibrated principal point for both photos in millimeters ( $X$  and  $Y$ ).
- (c) Radial lens distortion correction for both photos.
- (d) Digital image to photo transformations: the coefficients  $A1, A2, B1, B2, C1, C2$  in radians.
- (e) Exterior Orientation Parameters:  $OMEGA, PHI, KAPPA, XL, YL, ZL$  in radians and meters respectively.

These concepts are implemented into the subroutines PROJECT, XY2IJ and AFFIJ, these subroutines are called by the modules PTPROJ, PROJMBR, PROJEC, and SELREG with the purpose of obtaining the necessary interface between the 'geocoordinate' system and the image coordinate system.

## IV. THESIS DEVELOPMENT

### A. INTRODUCTION

To continue the study and development of the Image Interpretation System created by Prof. Chin-Hwa Lee and Prof. Hsi Jian Lee, a initial learning process was necessary. It included Prolog programming and an Image Processing course to be dictated by Prof. Chin-Hwa Lee, as well as general concepts in photogrammetry and some experience with the use of VAX/VMS system. FORTRAN programming was required to work with existing programs beside using Prolog. There was also one program written in Pascal.

The general orientation of the thesis was to improve the Image Interpretation System by reducing the CPU execution time, analyzing memory usage, and detecting possible software errors that could interfere with the process of Change Detection. It was suspected that some of the nodes being analyzed, were not properly detected by the Rule Base. If possible, additional modelling capabilities and different approaches to solve existing problems could be generated as part of the system's enhancement.

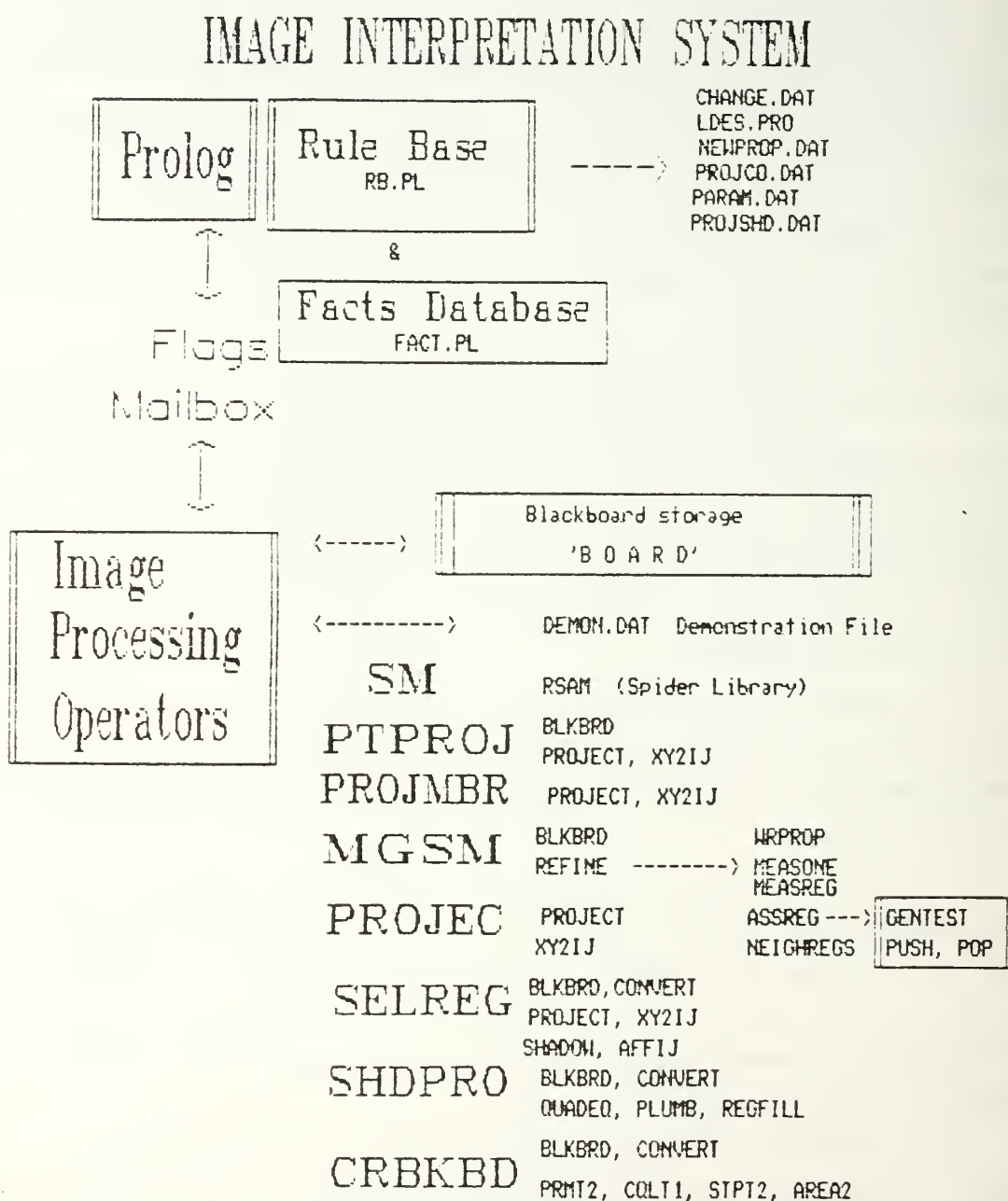
### B. SOFTWARE REVIEW

The study of the existing software was initiated with the review of the FACTS DATABASE and the RULE BASE. This study facilitated a general understanding of the problem solution approach as well as providing a grasp of how the 'model' parameters interacted.

The particular structure characteristics of the Prolog language, which include 'cut' operators, variable 'instantiations', and a 'backtracking' behavior, requires a lot of experience to analyze as in this case a complex system. The study of the Prolog part in the Image Interpretation System was more difficult than expected. additionally, the unavailability of source documentation at that moment made the task of reading and debugging more complicated.

The set of programs already created, see (Table 2), were developed in a time period of more than a year and half by Prof. Chin-Hwa Lee and Prof. Hsi Jian Lee. Because the topic is complicated, it required several weeks to understand the ideas behind every FORTRAN program and subroutine. Another fact that complicated the process of understanding was the invocation mechanism used by the Prolog Interpreter

TABLE 2  
SOFTWARE STRUCTURE





to initiate FORTRAN modules. This requires a protocol of setting flags to indicate initiation and completion status. The results of the FORTRAN modules were being transferred back to the Prolog Interpreter by means of 'mailbox' interprocess communication process, or by consultation of files generated by the FORTRAN modules. These files were formatted with the required mnemonics used in the facts database. The analysis of the FORTRAN modules was executed under schedule, and some syntax and logic errors were corrected. Concurrent to this analysis, the facts database was corrected by Prof. Chin-Hwa Lee, and the system exercised to observe the behavior under the new conditions.

To observe the results as the process goes through different stages in the interpretation process, two files are generated. The first is the CHANGE.DAT file, which provides the user with global results derived from the Prolog Rules. These are the certainty values used as evidence of the existence of a node. The file also shows some parameters used by the Change Detection Process as the image file name and region description file name. CHANGE.DAT is used as the output file in which the Prolog Interpreter responds to the entry command file called RB.COM. The last one is used as a batch command file to provide the system parameters and rule invocation commands.

The second file generated to show the results of the process is the file DEMON.DAT. This is a more detailed output that contains messages generated by each of the FORTRAN subroutines used along the main process. The information written here is used mainly for debug purposes because it keeps track of the regions modifications created by the Model Guided Split and Merge Module. Included in this file is information related to coordinates projected from earth system 3-D (latitude, longitude and height) to 2-D image coordinates. DEMON.DAT also presents the assigned properties to the new regions and the information of the shadow area generation.

The system can also generate a set of output files which can provide an insight into how the process, as a whole is carried out. These files are generated when running the system under debug mode 1. This is a more powerful mode and is used to understand the whole process, because it permits the graphic presentation of different regions as they are being created by the model guided split and merge process, the displayed images show the predicted shadow areas and projected model locations. These files are numbered sequentially and have the general form MSGnnn and



DEMnnn, where nnn is a sequence number beginning with 0. These files are used by the program HJTEST to display on the COMTAL system the labeled region map and other areas as indicated before.

By using these messages and demonstration files, is possible to understand the system behavior and detect possible errors in the process. It is necessary to mention here that the subroutine AFFIJ was created to shift the 'model' projection to its expected position. Due to errors in the collected data from the 'model', the projected vertices were not coincident with the corners of the buildings observed in the aerial photograph, some error was being introduced in the property results. The shadow properties were not exactly coincident and a 'affine' transformation was necessary to overlap the projected 'model' with the observed 2-D building projection in the aerial photograph.

After the 'affine' process was performed, a better region matching was obtained, and the shadow module produced improved results because the projected predicted areas were in position. However, 5 nodes were still not providing enough evidence to conclude their existence. Further study should be directed to determine the cause of this anomaly.

Because the amount of possibilities that could be presented to a system of this magnitude is enormous, the software review process still continues. However, as will be seen in later sections, the system performance was finally upgraded and all nodes were properly evaluated, increasing with it the percentage of system accuracy.

The software documentation has been written and is included as appendices at the end of the thesis .

### C. MEASURING THE CPU TIME

Another objective of the thesis was the verification of CPU time consumption while running the Image Interpretation System in different debug modes. It has been observed that the system performs an immense amount of calculations, which are needed by the Image Processing operators in the computational environment to analyze the labeled region files and generate new regions under the model guided split and merge module.

In order to have an efficient Image Interpretation System, the system should be able to analyze aerial photographs in reasonable periods of time. Depending on the complexity of the buildings under consideration, the CPU time consumption will also

TABLE 3  
CPU TIME CONSUMPTION (IN SECONDS)

PROGRAMS =====	MODE -->	DEBUG 0	DEBUG 1
CRBKBD		8.44	9.07
-----> Node 1 <-----			
SELREG		3.63	4.32
MGSM		3.26	6.32
-----> Node 2 <-----			
SELREG		3.80	4.39
MGSM		3.55	23.17
-----> Node 3 <-----			
SELREG		3.77	4.31
MGSM		6.24	63.66
-----> Node 4 <-----			
SELREG		3.65	4.35
MGSM		5.58	25.57
-----> Node 5 <-----			
SELREG		3.70	4.41
MGSM		7.17	79.73
-----> Node 6 <-----			
SELREG		3.74	4.30
MGSM		4.36	61.52
-----> Node 7 <-----			
SELREG		3.73	4.34
MGSM		5.19	24.27
-----> Node 8 <-----			
SELREG		3.70	4.39
MGSM		5.43	25.70
-----> Node 9 <-----			
SELREG		3.76	4.44
MGSM		(1)	(1)
-----> Node 10 <-----			
SELREG		3.68	4.36
MGSM		2.59	6.63
-----> Node 11 <-----			
SELREG		3.87	4.52
MGSM		(1)	(1)
-----> Node 12 <-----			
SELREG		4.01	4.70
MGSM		(1)	(1)
SHDPRO		70.69	93.43

(1) Not used module.

be different. For our experiment, we have used a 128 x 128 image, and the results are listed in Table 3 for a 'model' composed of 16 nodes. From the Table 3 it can be seen the great difference in CPU execution time consumption between DEBUG Mode 0 and

DEBUG Mode 1, which is caused primarily because the last processing mode generates a set of demonstration images and messages for everyone of the splited and merged regions, it also show in these images the general evolution of the segmentation process by displaying continuously the labeled region map.

Because there are intervals of time between the different process invocations, where the CPU have not been accounted by the program JCPU ( the CPU time measuring subroutine ), the total CPU time cannot be calculate precisely. However, the results obtained reflect reasonable estimates when considering the magnitude of the process.

## **D. INCORPORATING THE REFLECTANCE MODELLING**

### **1. Introduction (Image Irradiance and Surface Orientation)**

Having reviewed the system software, measured the CPU time performance, and corrected some errors to improve system performance, there were still some nodes in the 'model' that were not being detected properly. Several factors were influencing the evidence necessary to determine the existance of these nodes. One of the most critical factor was related to the 'certainty of tone'. The weight assigned to the property combined with the difference in the predicted and calculated tone was lowering the final average of certainty obtained by combining the certainties of size, polygon and shadow of the node. Therefore, the need to obtain a more accurate predicted tone was an obvious objective to be pursued.

An initial approach was suggested by the thesis Advisor to review some papers including an article on remote sensing, see (Ref 17), in order to incorporate reflectance capabilities to the Image Interpretation System. The task was to be carried out by the computational environment to perform the analysis and find out the effects of scene illumination over the node surfaces.

The problem of assigning a predicted tone for every node lies in the observation of surface normals, see (Refs. 18, 19), sun position and observer position. It is necessary to find out how they affect the determination of an accurate predicted tone compared to the calculated tone from the node projection over the digitized image and give a reasonable similarity measure.

Tone synthesis is based on a scene radiance equation that combines the bidirectional reflectance distribution function BRDF of the surface material, see (Ref. 17), the spatial and spectral distribution of light sources, and some assumptions about the atmospheric attenuation and path radiance.

## 2. Synthesis of the Tone

Unknown parameters of the 'model' have been obtained from the real image by sampling and experimentation. The results of comparison with the predicted tones and the calculated tones from the image will be obtained when the certainty of tone rule is executed by the Rule Base expert.

To synthesize the apparent brightness of a surface, we must consider the intrinsic reflectance characteristics of the surface material, the spatial and spectral distribution of the illumination, and the orientation of the surface relative to the observer and sun angle. The process uses the knowledge available at the time when the photograph was taken to simulate the relative angle of the light source. Also the reflectance factor of the surface (ALBEDO), which is unknown, is considered to be ideal for the effect of calculations and given a value of 1. We also assume that the surface materials forming the nodes are invariant with respect to the rotation of the surface normal, see (Fig. 4.1).

To calculate the radiance equation, we need to obtain the direction of the incident angle  $I$ , the emergent angle  $E$ , and the phase angle  $G$ . The incident angle  $I$  is the angle between the sun rays direction and the normal to the surface. The emergent angle  $E$  is the angle between the normal and the direction to the observer. The phase angle  $G$  is the angle between the incident and reflected rays. For a distant viewer and distant light source, as in the aerial photograph case, the phase angle  $G$  is constant.

To calculate these angles, we need to define the gradient of the node surface normal, and the gradient of the sun, which is the direction in the earth centered 'geocoordinate' system.

The BRDF (Bidirectional Reflectance Distribution Function) is an intrinsic property of a surface material. It determines how bright the surface will appear when viewed from a given direction and illuminated from another.

When illuminating a surface by a collimated source with irradiance  $E_0$  from a direction with gradient  $(P_0, Q_0)$  and by considering the illumination from the sky as a hemispherical uniform radiance  $E_S$  over the visible hemisphere, for the points not in shadow we get the following equation: see (Ref. 17).

$$Lr(I,E,G) = (E_0/P_0) * ALBEDO * \cos(I) + E_S * ALBEDO * ((1 + \cos(E))/2)$$

This implies the use of the emergent angle  $E$ , because differing surface elements 'see' differing amounts of the light source depending on surface slope. The ALBEDO is the reflectance factor of the surface considered ideal for the calculations with a value of 1.

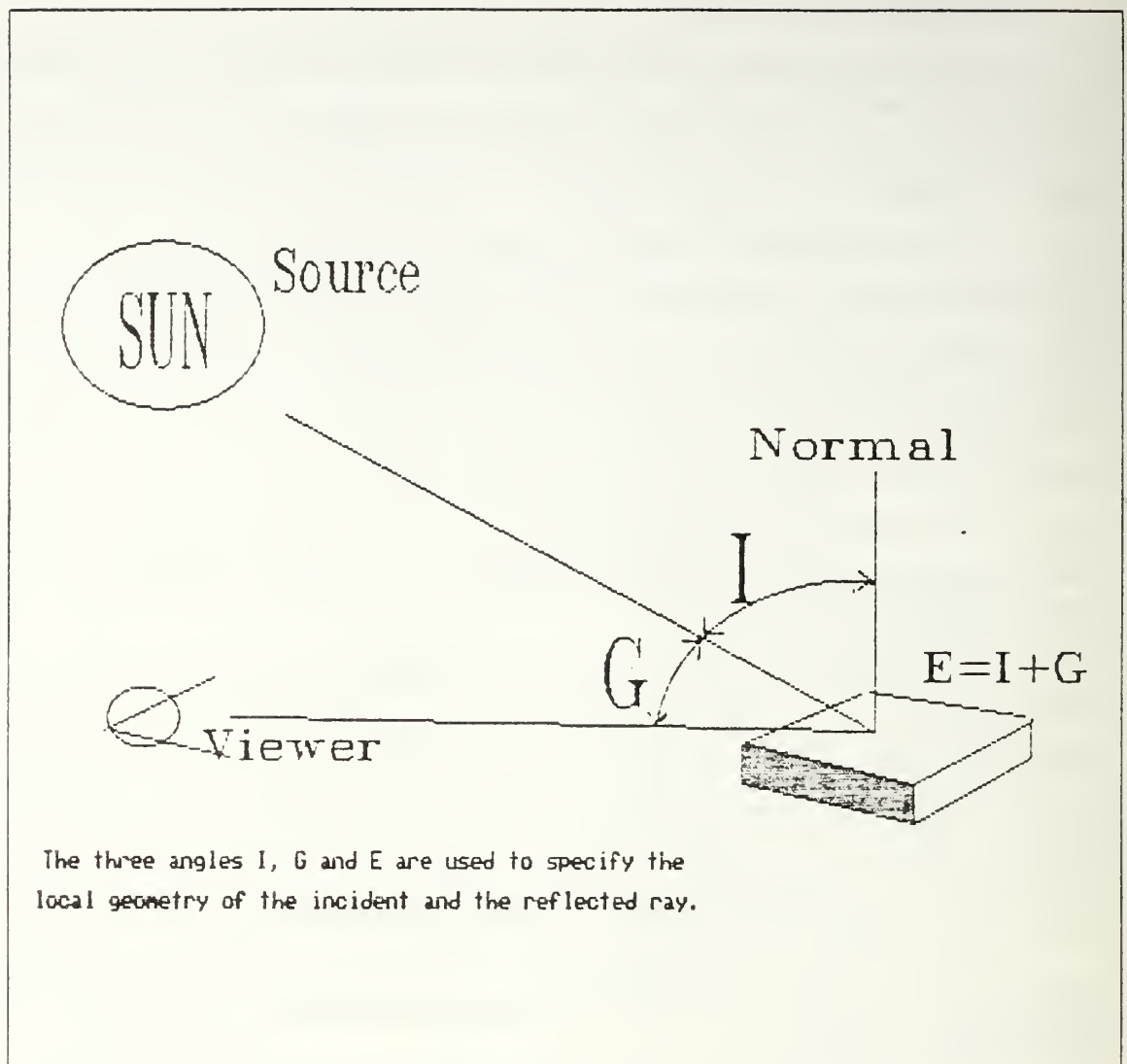


Figure 4.1 Local geometry of the incident and reflected rays.

The equation takes into consideration the gradient of the node surface normal, and at first approximation, the 'model' uses the sun as a distant collimated source, and the sky as a hemispherical uniform source.

The reflectance modelling program was created, and changes in the facts database were generated to accommodate the system to this new capability.

After several tries, the parameters used in the program were adapted to our 'model' and the system performance was upgraded. As result, all the previous nodes that were not providing enough evidence, now show strong conditions as to be evaluated and identified positively as existing and unchanged nodes.



The reflectance modelling was performed with all the different nodes, involving different tones according to the node's roof orientation, surface material category and image conditions. The results showed correspondence between the calculated values and the ones observed in the aerial photograph.

Further tone changes in the facts database showed clearly the sensitivity of the system to these changes. These experiments demonstrate the accuracy of the Image Interpretation System and the need of exercising the system under different conditions to calibrate the facts database to its optimum values.

## V. CONCLUSIONS

Image Interpretation of Aerial Photographs is a difficult task considering the vast amount of calculations involved in the analysis of image arrays. By using a computer, in this specific case a VAX 11/780 running under the VMS operating system, and using an Expert System to 'minic' the human reasoning it has been demonstrated that the process of detecting changes in buildings from an aerial photograph is a reality. Even more, this Expert System, which has been created under a firm base, could be able to adapt to future expansions, or to be implemented in other Prolog environments.

The ideas behind the system design are solid, promising and ambitious, the results are reliable and effective within the actual technology constraints. The objectives that were delineated in the initial statement of the investigation have been accomplished in the following way:

1. The in-depth analysis of all the Fortran programs and subroutines conforming the image processing operators and the Prolog facts database and Rule Base have been performed. Some logical and syntactical errors were found and corrected. The documentation of both environments, the Prolog and the image processing programs, was generated and are included as Appendices A and B.
2. The facts database was modified to accept the new reflectance capabilities introduced to the system. The changes were made according to experimental data sampled from the aerial photographs. The Rule Base was changed in the same sense, but only minor changes were generated. These are located in the 'corresponding regions' rule and are designed to receive the incoming results from the selecting regions module, which are originally created by the reflectance program.
3. The CPU time was measured, and it showed a distributed consumption over the different image processing operators and prolog area. The values are within a reasonable margin and didn't show any excessive time consumption in the Image Interpretation system.
4. The process of exercising the system with different kinds of photographs has been partially simulated by using a modified 'model'. The results show great sensitivity of the system to changes in tonality and position of the nodes in the 'model'.
5. The reflectance modelling capabilities were developed and implemented in the Image Interpretation System. The overall results present higher accuracy in the output from the Rule Base. There is clear evidence that indicates the progress obtained in the conclusions of matching the 'model' to the aerial photograph. This implies more confidence in the system productivity.

The actual Image Interpretation System could be transformed into a faster, more reliable, and flexible system by the use of a more powerful Prolog compiler such as QUINTUS Prolog. The ideas used to develop this system are well conceived, the results are optimistic and the system could be enhanced in the future to generate its own rules and adapt more easily to a given situation.

## APPENDIX A

### PROLOG ENVIRONMENT SUMMARY

#### 1. RULE BASE EXPERT

##### a. Functions performed

Given a specific environmental 'model' and a defined camera conditions, find the changes in the digitized image taken from the aerial photograph.

##### b. Definitions

The certainty of Node N is:  $certainty\_children(N) * certainty\_property(N)$ , where *certainty\_children* is the arithmetic average of the certainty values of all children of the node, and *certainty\_property* is the weighted sum of its properties; *checkinglist* is one of the dynamic databases.

##### c. Program structure

The following steps are accomplished by the process in sequence:

(a) Get the following input parameters from the user (example data):

Labeled region file:	Labfile (LLAB.DAT).
Image file:	Imgfile (L220200.DAT).
Side of the image:	Side as (LEFT).
Debug Mode:	Debug (0/1/2).
Image Identification:	ImageId (LEFT_PICT1).
Prolog region description file:	Desfile as LDES.PRO.

(b) Write the parameters in the file PARAM.DAT as well as the affine coefficients for the respective side of the stereo pair of images.

(c) Create the blackboard interprocess communication area by initiating the fortran program CRBKBD with the statement: 'Simage' (crbkbd, {}, X ).

(d) The Prolog expert must consult the region description file to get all the region properties. Continue reading the entry command file RB.COM and proceed with the subsequent control rules.

##### d. Called routines

CRBKBD: Creation of the 'blackboard' for interprocess communication.

##### e. Other considerations

None.

## 2. FACTS DATABASE

### a. Functions performed

The prolog file FACT.PL serves as the environmental 'model'; it has a 'tree' like structure and it is the static database of the Change Detection System. its tree structure is composed of a number of nodes, and each node has a unique name with almost similar format. The environmental 'model' has the following characteristics:

- (a) The hierarchical relationship in the tree is fixed; the hierarchy from top to bottom of the building family is as follows: block, building cluster, building, building parts and subparts.
- (b) Property inheritance: By this property, the system extracts the missing properties of a node from the ones of its ancestors.

### b. Definitions as used in the Prolog expert

coefficient:	Affine transform coefficients.
smc_tone:	Surface Material Category.
shadow_smc_tone:	Variation of smc tone used for shadow areas.
block:	A collection of tree roots.
type:	Defines the classification of hierarchies.
class:	Describes the type of environmental 'model'.
children:	Are the roots originated in the node.
from:	Relates a node to its ancestors.
mbr:	Is the Minimum Bounding Rectangular area enclosing a node.
background_smc:	The background smc surrounding the node base plane.
context:	Gives a written statement about the node identification.
compact:	Gives a simple meaning of the general shape of the node.
no_vertex:	The number of vertices in the node polygon.
polygon:	Describes the 3-D coordinates of each vertex in the node.
height:	Provides the node's height.
property:	Gives the properties and weights that should be analyzed for a given node.
ref_point:	Reference point used to observe the sensed image light conditions.
predicted_area:	Opposite corner coordinates of the the rectangle enclosing the block used to detect if the sensed image holds the area of interest for the change detection process.

c. **Called routines**

None.

d. **Other considerations**

None.

3. **CONTROL RULE #1**

a. **Functions performed**

Find the reference point ratio in order to modify the calculations of the change detection process accordingly to the light conditions on the image. Use the *find\_ref\_ratio* rule.

b. **Definitions**

None.

c. **Program structure**

The following steps will be accomplished by the process:

- (a) Get from the facts database the reference point coordinates. Obtain the 2-D projection coordinates of the reference point; use the average value of the surrounding pixels in the image to get a more accurate reference.
- (b) Compare to the facts database reference value and establish a ratio that will be used through the whole change detection process.

d. **Called routines**

The program PTPROJ is initiated with the sentence: 'Simage' ( ptproj, Coord, List).

e. **Other considerations**

None.

4. **CONTROL RULE #2**

a. **Functions performed**

Mark a block as a root node for subsequent change detection process if the intersection area of it and a segmented image is large enough.

b. **Definitions**

None.

c. **Program structure**

The following steps will be accomplished by the process:

- (a) Select a block from the facts database, use *type (X,block)*, and get its MBR before initiating the fortran program PROJMBR; once we get back the 2-D image projection, use the SIDE parameter to extract the X and Y coordinates of the



projection of the block, for the left or right side of the image stereo pairs use the *routine\_project\_mbr* and *separate\_project\_mbr* rules.

- (b) Obtain the image file identification and use it to extract the predicted area of the block from the facts database. Use the coordinates of the predicted area and the ones provided by the PROJMBR program to obtain the area of intersection of both MBRs; use the *intersection\_betw\_mbrs* rule.
- (c) Get the size of the projected mbr and the size of the intersection part, use the *mbr\_size* rule; obtain the ratio by dividing the 'intersection MBR' size over the 'projected MBR' size; use the *ratio (Size\_Ints, Size\_MBR2, R)* rule.
- (d) Check the ratio of intersection: if the block is identified to be in the image, its ratio should be higher and the change detection process could be initiated; use the *check\_block* rule; the ratio must be over 60 per cent in order to determine that the block is positively located the image.
- (e) By the same ideas, locate the building cluster from the block, get the building from the cluster and use the ratio of MBR's size intersection before checking the building in order to be sure that is properly located in the sensed image; when the building terminal nodes are detected, add them to the checking list, which will be constructed as a list of all the elements to be analyzed along the change detection process. Use the following rules: *find\_building\_cluster\_intersection*, *find\_intersection\_a\_building*, *check\_building* and *put\_node* to locate and add the node to the checking list.

**d. Called routines**

PROJMBR: initiated by using the statement 'Simage' (project\_mbr, MBR, Proj\_MBR).

**e. Other considerations**

None.

**5. CONTROL RULE #3**

**a. Functions performed**

Activate one of the nodes in the 'checkinglist' dynamic database.

**b. Definitions**

None.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) Check if it is the last node in the 'checkinglist' ( the last one must be equal to "markL" ) by using the rule *last\_node\_test (Node)*.
- (b) If it is not the last node, validate it through the analysis process; use the control rule 4: *validate(Node)* and retract it from the 'checkinglist' when finished.

**d. Called routines**

None.

**e. Other considerations**

None.

**6. CONTROL RULE #4**

**a. Functions performed**

If the node is a non-terminal node, then add its children to the 'checkinglist'.

**b. Definitions**

None.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) Get the children list of the node from the facts database and use the *put\_node* rule to initialize its certainty value to -9999.
- (b) Include the node's children in the 'checkinglist'; use the *validate (Node)* rule.

**d. Called routines**

None.

**e. Other considerations**

None.

**7. CONTROL RULE #5**

**a. Functions performed**

If node is a terminal node, and its certainty value is undecided, then calculate its certainty value by combining all the property's values obtained; update its father certainty and eliminate it from the checkinglist.

**b. Definitions**

None.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) Calculate the property certainty of the node by first initializing the property certainty value to -9999; get the list of properties and its weights for the specified node from the facts database; obtain the combined certainty value for all the properties, by multiplying the weights of the individual properties by its certainty factors and dividing the result over the sum of all the weights, use the *validate(Node)* rule.
- (b) Report its status (see control rule 10).
- (c) Refine the results by the 'model' guided split and merge program; if the results are not satisfactory (see rule No. 11).
- (d) Update the father certainty of the node (see rule No. 8).
- (e) Remove the node from the 'checkinglist' (see rule No. 6).

**d. Called routines**

None.

**e. Other considerations**

None.

**8. CONTROL RULE #8**

**a. Functions performed**

If the certainty values of the node's children are completely decided, then update the certainty value of the node's father.

**b. Definitions**

None.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) Obtain from the facts database the list of all children corresponding to a node's father. use the *from (Node, Father)* and *children ( Father, L)* facts.
- (b) Get all the children's certainty value; if one of them have not been solved yet, return; use the *all\_children\_solved ( L, Children.No)* rule.
- (c) Combine the children certainty factors and get the average by dividing the total sum of children's certainty over the number of children, use the *combine\_certainty\_of\_children ( L, Children\_No, X1)* rule.

- (d) Find the father's property certainty; in case that there are particular properties to find, like 'shadow', use the *find\_father\_property\_certainty* (*Father*, *Value*) and *calculate\_property\_certainty* (*Father*) rules, as seen in control rule No.5 part (a).
- (e) Combine the children and father certainties; if one of them is equal to -9999 or 0, use the other to obtain the combined value, use the *combine\_children\_and\_property\_certainty* rule.
- (f) Replace the 'instantiated' value of -9999, report its status and repeat the process for the upper hierarchies.

d. Called routines

None.

e. Other considerations

None.

## 9. CONTROL RULE #9

a. Functions performed

If the certainty value of a node is decided, then remove the node and its successors from the 'checkinglist'.

b. Definitions

None.

c. Program structure

The following steps will be accomplished by the process:

- (a) Get the node's children list and remove the list from the facts database; use the *remove\_from\_checkinglist* and *remove\_children* rules.

d. Called routines

None.

e. Other considerations

None.

## 10. CONTROL RULE #10

a. Functions performed

If the certainty value of a node is decided, then report its status.

b. Definitions

None.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) Get the node's certainty value from the facts database using the *certainty\_value* ( *Node*, *Value*) fact; if the value is -9999 return.
- (b) Check the value against certain limits to decide different conclusions; write the results in the CHANGE.DAT report file, use the *status* ( *Node*, *Value*) rule.

**d. Called routines**

None.

**e. Other considerations**

None.

**11. CONTROL RULE #11**

**a. Functions performed**

Split and merge: If the certainty of a node cannot be decided after all properties are evaluated, then try to split and merge the regions associated with the node (these are the corresponding regions obtained by the *corresponding\_regions* rule).

**b. Definitions**

None.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) If the certainty value of the node is already greater than 60 per cent, return.
- (b) If not, get the *possible\_regions* list that was generated by the corresponding regions rule and call the MGSM program to reconfigure the regions intersecting the node's projection, in order to evaluate more precisely the different property certainty factors, use the *possible\_regions* ( *Node*, *RegList*) fact.
- (c) The facts database is modified according to the results produced by the MGSM program, these results include the new region properties in the file NEWPROP.DAT, use the *update\_a\_record* ( *Term*) rule.
- (d) Calculate the property certainties as was done in the control rule No. 5 and report the new status, use the *calculate\_property\_certainty* ( *Node*) rule.

**d. Called routines**

MGSM: initiated by the sentence 'Simage' (mgsm, RegList, X).



e. Other considerations

None.

12. CERTAINTY OF TONE

a. Functions performed

The certainty of tone is assigned according to the average value of the tone difference of all the corresponding regions.

b. Definitions

None.

c. Program structure

The following steps will be accomplished by the process:

- (a) Get the list of corresponding regions to the node's projection; use the *corresponding\_regions* rule.
- (b) Get the surface material category of the node from the facts database and multiply it by the ratio obtained in control rule No. 1, this is the node relative gray level.
- (c) Obtain the difference of tone in each one of the corresponding regions, with the relative gray level of the node being analyzed. Get the list of tone differences and sizes of each tone included in the corresponding regions list, use the *sub\_tone\_diff* and *list\_of\_tone\_diff* rules.
- (d) Obtain the weighted average by multiplying and summing the tone and size of each one of the corresponding regions over the sum of the intersected region's area, use the *weighted\_sum* rule.
- (e) Assign a tone certainty value according to the weighted average obtained, use the *assign\_value\_tone* rule.

d. Called routines

None.

e. Other considerations

None.

13. CERTAINTY OF SIZE

a. Functions performed

Assign a certainty value according to the ratio between the node size and the sum of all the corresponding region's sizes intersecting the projected node.

**b. Definitions**

None.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) Get the list of the corresponding regions and their sizes, get the mapped size of the node from the facts database generated by the *corresponding\_regions* rule.
- (b) Assign a value according to the ratio between the mapped size and the sum of the list. use the *RATIO* fact and *assign\_value\_size* rule, to generate the certainty value of this rule.

**d. Called routines**

None.

**e. Other considerations**

The *list\_of\_region\_size* rule gets from the facts database the sizes of all the corresponding regions. The *RATIO* is defined as mapped node size over sum of the corresponding regions; the last predicate is obtained through the *sum\_list* rule.

**14. CERTAINTY OF SHADOW**

**a. Functions performed**

Assign a value to the shadow certainty according to the predicted and calculated shadow areas projected from the node.

**b. Definitions**

The property inheritance: a concept used when the property of a node is not found, it is searched through its ancestors; to obtain it use the *find\_background\_smc* rule.

**c. Program structure**

The following steps will be accomplished by the process:

- (a) Obtain the node's descendants: get the node's children from the facts database, and with this information, create a table of possible descendants initializing the last element to "markL" ; continue searching for new generations until all descendants are collected; use the *find\_next\_generation*, *collect\_descendant* and *getnext\_des (DES)* rules.
- (b) Generate the shadow projected polygons: for every descendant node get the number of vertices forming the polygon and extract its coordinates to be written into the file *PROJSH.DAT*; use the *form\_proj\_polygons* and *write\_proj\_coord*

rules. Note that the projected shadow coordinates were originally generated by the 'corresponding regions' rule, which called the SELREG program, and wrote them in the PROJCO.DAT file.

- (c) Initiate the shadow processing program SHDPRO with the statement 'Simage' (shdpro, {}, L).
- (d) After the program SHDPRO generates a list of area-tone pairs, these are the different regions intersecting the projected shadow areas; obtain the average wighted tone value after summing the weighted tones and dividing proportionally to their areas; use the *weighted\_sum\_shadow* rule.
- (e) Assign a value to the shadow certainty, according to the absolute difference between the predicted tone of the projected shadow, and the calculated tone of the sum of the intersecting regions taken from the image file; use the *assign\_value\_shadow* and *certainty\_value shadow* rules.

d. Called routines

SHDPRO: Shadow processing program.

e. Other considerations

None.

## 15. CERTAINTY OF POLYGON

a. Functions performed

Generates a certainty value according to the ratio between the intersecting parts of the regions overlapping a node and the mapped size of it.

b. Definitions

None.

c. Program structure

The following steps will be accomplished by the process:

- (a) Get the mapped size of the node from the facts database; obtain also the intersected area's size of the different regions by using the *mapped\_size* and *intersect\_area* facts; add the list of intersection areas and obtain the ratio by dividing it over the mapped size; use the *sum\_list* rule; assign a certainty value according to the ratio; use the *assign\_value\_polygon* rule.

d. Called routines

None.

e. Other considerations

None.

16. PROLOG COMMON ROUTINES

a. Functions performed

Provide information for several properties and rules through the change detection process.

b. Corresponding regions rule:

The purpose of this rule is to find the regions related with a specific node; it produces the following facts and tables to be used by the Interpreter: mapped size of the node, list of regions corresponding to a node area and areas of intersection, also generates through program SELREG, and the projected and shadow predicted coordinates of the node vertices. The following steps will be accomplished by the process:

- (a) Check if the possible regions table already exists; return if so. If not create a vertex file called VERTEX.DAT to be used by program SELREG call it by the command 'Simage' (selreg, {}, M).
- (b) From the list received through mailbox created at the conclusion of SELREG, generate the list of intersecting areas and possible regions overlapping the projected node; use the *collect\_regions* and *region\_number\_and\_intersec\_size* rules.

c. Mathematical common routines.

The mathematical common routines are used to find differences, min, max and absolute values.

d. Utility common routines

Are used to find the minimum number in a list as *min\_list*, and number of elements in a list as *no\_elements* rule.

e. Called routines

None.

f. Other considerations

None.

## APPENDIX B

### COMPUTATIONAL ENVIRONMENT SUMMARY.

#### 1. PROGRAM (FORTRAN) 'SELREG'

##### a. Functions performed

Select a list of regions intersecting with the area projected by a node from the environmental 'model' and get a list of the image projection coordinates in 2-D corresponding to the vertices of the node and their shadows. This information will be written into a file to be used by the shadow verification subroutine. Additionally, this program will handle the modification of the facts database according to new tone values generated by to the sun position relative to the node surface normal.

##### b. Input

NODENAME:	The node being analized.
ITONE:	The node tone before sun illumination analysis.
COORD:	The earth coordinates X,Y,Z of every node vertex.
IN_ARGS:	The number of arguments received through mailbox.
NOVERTEX:	The number of vertices for each node.

##### c. Output

VERTEX:	The 2-dimension coordinates of each vertex projection.
IS, IJ:	The 2-dimension coordinates of each vertex shadow projection.
ITONE:	The modified value of the tone for each node.
NOREG:	The region intersecting the node projection.
NOPOINTS:	The number of pixels related to the intersection area.
OBUF:	The output buffer to be sent through mailbox.
NARGS_OUT:	The number of arguments being send out.

##### d. Common area

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

MODEL:	This matrix provides the 'model' 2-D projection.
REGMAP:	Includes all the segmented regions.
LR:	Side of the stereo pair of images.
SUNANG:	Provides with the absolute sun direction.
PLNCOEF:	Are the base plane coefficients A, B, C and D.
FILESEQ:	A value used to identify the progress report files.



COEF:	Coordinates corresponding to the exposure station.
NREGS:	Is the number of regions identified in the photograph.
AREA:	The region's area.
PERI:	The region's perimeter.
COMPACT:	Parameter that show a generated shape relation of the region.
MTONE:	The average tone of each region.
VTONE:	The tone variance of every region.
MBR:	The minimum boundary rectangle for every region.
DEBUG:	The type of debug process under execution (0,1 or 2).
IMAGE:	The digitized image of the aerial photograph.

e. **Work area**

NCHARS:	The number of characters in the node name.
MBR2D:	The MBR of every node projection.
STRBUF:	A buffer to hold the input data for the program.
INUM:	Number of elements returned by subroutine CONVERT.
U, V, W:	Real values for X, Y and Z.
COUNTX,1:	Variables to hold the CPU time consumed by the program.
ROOF:	Matrix to hold the real values of the node's vertices.
NO_REGS:	The number of regions intersecting the node projection.
RATIO:	The relation of the intersected part to the 'model' size.
NO_MODEL:	Number of pixels in the 'model'.
MBRINT:	The MBR of the intersection part for every corresponding region.

f. **Parameters and modes**

This program is designed to work with debug modes 0 and 1 (mode 2 is under development), when running in mode 1 there is additional information generated which will be displayed in the COMTAL environment to show the process progress report. The program uses as parameters, ISX and ISY with values of 128 which are the ranges in X and Y of the aerial photograph; MAXREG is 100 and it is the limit of regions to be obtained from the segmented image; finally the left side of the stereo pair is assigned as a value of 1 and the right one as 2.

g. **File handling**

DEMON.DAT:	Output file that contains all process messages.
VERTEX.DAT:	Input file created by the change detection process in order to provide with the input parameters.

PROJCO.DAT: Output file to be consulted by the shadow processing subroutine, it include projected and shadow coordinates.

#### h. Program structure

The program for region selection is summarized in the following steps:

- (a) The selecting regions program is initiated by the Prolog Interpreter, and the first task is to establish communication with the memory management interprocess routines: event flag association, mailbox communication and logical unit association.
- (b) The file DEMON.DAT is opened to receive all the information regarding the development of the process and the file VERTEX.DAT is opened to provide all the input parameters; next all the work area variables are initialized.
- (c) Immediately for every vertex are calculated its 2-D projected and shadow coordinates and also generated the minimum boundary rectangular of the node. The new tone for the node is then calculated with all the information relating the plane equations of the different node vertex combinations.
- (d) The MODEL matrix is progressively being generated with all the accumulated 2-D projection of the different nodes, where the previous processed nodes are represented by code '2' and the last node by '1'; the last node being processed is retained in the matrix BUF2D to be used for intersection calculations.
- (e) Finally all the intersecting regions and their intersection area will be transferred back to the calling program via mailbox interprocess communication.

#### i. Exception handling

Not used.

#### j. Calling routines

This program is initiated by the Prolog Interpreter during execution of the statement "Simage' (selreg, {}, M)" in the "corresponding-regions" rule.

#### k. Called routines

JCPU:	Used to calculate the CPU time.
BLCKBRD:	To obtain connection with the blackboard storage.
AFFIJ:	Generate additional affine transformation to the image.
CONVERT:	To transform the ascii to integer values.
PROJECT:	To project a 3-D coordinate to 2-D coordinate.
XY2IJ:	Perform the affine transformations of the image.
SHADOW:	Used to generate the shadow projections.
REFLEC:	To calculate the tone variation of the node.

REGFILL: To fill the intersected part of a region and node.  
FILENAM: Used to provide a different filename for the progress  
report process while working in mode 1.

l. **Timing constraints**

Not observed.

m. **Other considerations**

- (a) In order to link this program, the following command will have to be given:  
Slink selreg, affij, blkbrd, shadow, convert, ufo\_create, ufo\_open, filenam, jcpur,  
reflec, projmbr, projaff, regfill, blkbrd/opt.
- (b) The changes performed to the program are related to the insertion of the timing  
routines, and with the generation of the new tone for the different nodes  
according to the sun position.

2. **PROGRAM (FORTRAN) 'MGSM'**

a. **Functions performed**

Is the main program of the 'model' guided split and merge module.

b. **Input**

IN\_ARGS: The number of arguments received through mailbox.

REGLIST: List generated by corresponding regions rule.

c. **Output**

The output of this program is transferred to the Prolog Expert through  
mailbox.

d. **Common area**

The common area called BOARD has been developed to provide interprocess  
communication through a background storage and includes:

MODEL: This matrix provides the 'model' 2-D projection.

REGMAP: Includes all the segmented regions.

LR: Side of the stereo pair of images.

SUNANG: Provides with the absolute sun direction.

PLNCOEF: Are the base plane coefficients A, B, C and D.

FILESEQ: A value used to identify the progress report files.

COEF: Coordinates corresponding to the exposure station.

NREGS: Is the number of regions identified in the photograph.

AREA: The region's area.

PERI: The region's perimeter.

COMPACT:	Parameter that show a generated shape relation of the region.
MTONE:	The average tone of each region.
VTONE:	The tone variance of every region.
MBR:	The minimum boundary rectangle for every region.
DEBUG:	The type of debug process under execution (0,1 or 2).
IMAGE:	The digitized image of the aerial photograph.

e. **Work area**

Not used

f. **Parameters and modes**

This program is designed to work with debug modes 0 and 1 (mode 2 is under development), when running in mode 1 there is additional information generated which will be displayed in the COMTAL environment to show the process progress report. The program uses as parameters, ISX and ISY with values of 128 which are the ranges in X and Y of the aerial photograph; MAXREG is 100 and it is the limit of regions to be obtained from the segmented image; finally the left side of the stereo pair is assigned as a value of 1 and the right one as 2.

g. **File handling**

DEMON.DAT: Output file that contains all process messages.

h. **Program structure**

The program for split and merge is summarized in the following steps:

- (a) This is the main program of the module, is in charge of handling the interprocess communication and passing parameters to the subroutine REFINE, which practically performs all the splitting and merging tasks.

i. **Exception handling**

Not used.

j. **Calling routines**

This program is initiated by the Prolog Interpreter during execution of the statement "'Simage' (imgsm, {}, M)" in the control rule No. 11.

k. **Called routines**

REFINE: To refine the region's association with the node.

l. **Timing constraints**

Not observed.

m. Other considerations

- (a) In order to link this program, the following command will have to be given:  
Slink mgsm, refine, assreg, neighreg, tone, measone, measreg, wrprop, jcpur,  
filenam, ufo\_create, ufo\_open, blkbrd, npsSlibrary:spiderlib lib

3. PROGRAM (FORTRAN) 'SM'

a. Functions performed

To segment an image based on an split and merge algorithm. The program uses the RSAM segmentation process which is one of the SPIDER library subroutines.

b. Input

Input data is entered interactively by the user.

c. Output

A tone data file, and a region labeled file are generated.

d. Common area

Not used.

e. Work area

FNAME:	The image file name as L220200.DAT.
OUTFILE:	The output labeled region file as LLAB.DAT.
INBUF:	Input buffer to hold each input file record.
IP:	The image data in integer format, values from 0 to 255.
MAP:	Character codes to label the segmented regions.
L0:	Level at which segmentation starts (default = 3).
LN:	Logarithm to base 2 of ISX and ISY = 7 (for ISX = 128).
JERR:	Return error code.
JP:	Output labeled buffer.
NP:	Index of labeling codes used with "MAP".
ITHV:	Maximum difference for merging and splitting a region, the default value is 20 (over 256 possible max).
NTAREA:	Area threshold for small region elimination performed after each region grouping (default = ISX*ISY/200).
NTINT(ntarea):	One dimensional array in which threshold of difference in image intensities is stored as function of area in small region elimination (def. values are 50 to 30).



**f. Parameters and modes**

A noise region is defined as  $ISX * ISY / 1000$ .

**g. File handling**

FNAME:            Input file with the digitized image data.  
OUTFILE:          Output file with the labeled regions.  
TONE.DAT:        Output file with the tones of all labeled regions.

**h. Program structure**

The following steps will be accomplished by the segmentation process:

- (a) Get the input parameters : The image file name, output labeled region file and prepare them to process the input/output data, also the file TONE.DAT is used to place all the tone information related to the different regions.
- (b) For diagnostic purposes, the process display the multiples of 16 No.s of records as they are being read from the input file; load the image data after transforming it from byte format to integer format with values varying from 0 to 255.
- (c) Generate the parameter and arrays as required by RSAM; the actual values used are: ITHV = 70, L0 = 4, NTAREA = 120, NTINT(1)= 70, NTINT(ntarea) = 70, then call the subroutine RSAM, and after return check for error codes.
- (d) Check the contents of every region generated, and decide if it is too small to be considered as noise; assign to it one of the possible ascii codes.
- (e) Obtain the tone of the region and write it in the tone file to be consulted by the Prolog Interpreter.

**i. Exception handling**

This program checks for error codes generated by RSAM; check if the region number exceeds 200, which is the maximum value; and provide with error messages.

**j. Calling routines**

This program is a preparatory step previous to the invocation of the Change Detection System.

**k. Called routines**

RSAM:            Segmentation routine from the Spider Library.

**l. Timing constraints**

Not observed.

**m. Other considerations**

None.

#### 4. PROGRAM (FORTRAN) 'SHDPRO'

##### a. Functions performed

This program will generate: a list of regions intersecting the projected shadow area, and their mean tone values: to do it, this subroutine will use the vertices of the node and the projected shadow points that were generated by the subroutine SHADOW; this list of regions and tones will be used by the shadow certainty rule of the Change Detection System.

##### b. Input

IN\_ARGS:           The number of arguments received through mailbox.

NPS:               The number of vertices in a polygon.

Note that the received coordinate buffers are composed of 4 elements, the order used is: 2-D coordinates of the projected image and 2-D coordinates of the predicted shadow points.

##### c. Output

The output parameters are send out via mailbox interprocess communication scheme.

ITONE:            The average gray value of the intersection area.

NPOINTS:          The number of pixels related to the intersection area.

OBUF:             The output buffer to be sent through mailbox.

NARGS\_OUT:        The number of arguments being send out.

##### d. Common area

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

MODEL:            This matrix provides the 'model' 2-D projection.

REGMAP:           Includes all the segmented regions.

LR:               Side of the stereo pair of images.

SUNANG:           Provides with the absolute sun direction.

PLNCOEF:          Are the base plane coefficients A, B, C and D.

FILESEQ:          A value used to identify the progress report files.

COEF:             Coordinates corresponding to the exposure station.

NREGS:            Is the number of regions identified in the photograph.

AREA:             The region's area.

PERI:             The region's perimeter.

COMPACT:          Parameter that show a generated shape relation of the region.

MTONE:            The average tone of each region.

VTONE:	The tone variance of every region.
MBR:	The minimum boundary rectangle for every region.
DEBUG:	The type of debug process under execution (0,1 or 2).
IMAGE:	The digitized image of the aerial photograph.
<b>e. Work area</b>	
STRBUF:	A buffer to hold the input data for the program.
NPS_SHD:	The number of points in the calculated shadow area.
INUM:	Number of elements returned by subroutine CONVERT.
MBR_QUAD:	Is the MBR that holds the QUADLTL polygon.
PREDREG:	Buffer to hold the total calculated shadow area.
POLYGON:	Buffer to hold the coordinates that form the projected node.
MBR2D:	Hold the projected vertices and the shadow projection MBR area generated by shadow process.
FLAG1,2:	Are logical values generated by the logical function PLUMB, and indicate whether a point is in or out a polygon.
NO_REGS:	The number of regions selected as candidate shadow projection regions.
NP_PREDREG:	The number of points in the total calculated shadow area generated by the polygon.
VERTEX:	It holds the projected edge coordinates and predicted shadow coordinates from a polygon.
POINT:	Temporary area used by subroutine CONVERT to transfer its output results.
CP:	Buffer to hold the central points of the lines joining adjacent edge coordinates or shadow projection points.
XSOL1,2:	Coordinates in X at a distance DIST of the central point as explained in CP.
YSOL1,2	Coordinates in Y at a distance DIST of the central point as explained in CP.
A, B, C:	Are the line coefficients of the line joining two adjacent projected vertices or predicted shadow coordinates.
QUADLTL:	Is an area formed by the predicted shadow coordinates and the node's edge projection points that originated them.
NTEST:	Is the number of points that have been generated in the predicted shadow projection process.

J,INEIGH:	Offset to enlarge 1 pixel around the total calculated shadow area.
MBR_INT:	MBR to hold the intersection of the shadow area with a specified region.
RATIO:	Is the number of points in the shadow intersection to the number of points in the total shadow ratio.
BUF2D:	Buffer to hold the calculated shadow projection from a node's edge.

#### f. Parameters and modes

As parameters used by this subroutine, we find the following: DIST is the distance from a calculated point perpendicular to the central point in a line joining two vertices, this is initialized to 0.1; MIN\_RATIO: is the value used to determine if the intersected part of a shadow and a specific region is large enough, this value is initialized to 0.1; and MAX\_BOUNDARY, is the number of points necessary to trigger the analysis of a specific region adjacent to a shadow area, it is initialized to 3.

This program is designed to work with debug modes 0 and 1 (mode 2 is under development), when running in mode 1 there is additional information generated which will be displayed in the COMTAL environment to show the process progress report. The program uses as parameters, ISX and ISY with values of 128 which are the ranges in X and Y of the aerial photograph; MAXREG is 100 and it is the limit of regions to be obtained from the segmented image.

#### g. File handling

DEMON.DAT: Output file that contains all process messages.

PROJSHD.DAT: Input file that contains the projected vertices coordinates and the predicted shadow coordinates.

#### h. Program structure

The program for shadow generation is summarized in the following steps:

- (a) Initiate interprocess communication through event flag association, mailbox and logical unit selection; open the files DEMON.DAT and PROJSHD.DAT in order to write all the development messages, and to get the input parameters. Finally the work area is initialized by clearing the buffers to hold the predicted shadow region and MBR2D is initialized.
- (b) The input parameters (2-D vertex coordinates and predicted shadow projection) are loaded into the program, and the MBR that holds the total node projection is generated.



- (c) The projected shadow is generated by first calculating the central points of both predicted shadow projection sides, and projected edge sides; this is performed by finding the point in a vertical line that is perpendicular to the line connecting two adjacent vertices. If this connecting line is not parallel to one of the axis (X or Y), then the slope and the intersecting points will have to be used in order to get the line coefficients and by this, obtain the central points.
- (d) Once the possible solutions are obtained by offseting the central points to both sides of the connecting line by a distance DIST, then the logical function PLUMB will provide with the logical values FLAG1 and FLAG2, which are the indicators to decide if the possible solution lies inside or outside the polygon formed by the vertices and the shadow projection points.
- (e) By choosing the solutions that lie inside the polygon formed by the predicted shadow points, it is possible to decide if one of the sides of the node is generating a real shadow, the method used is to verify if a central point of an edge, and the interior point of the shadow polygon lie in the same side of a line connecting two adjacent shadow projected points.
- (f) A MBR called MBR2D that was holding the node projection, now is also enlarged to hold the calculated shadow projection; The calculated shadow that is underimposed by the node projection is eliminated from the predicted region area. In case of working under mode 1, the shadow shapes projected from each side of the polygon are transferred to demonstration files.
- (g) Now, the shadow shape is enlarged by one pixel in order to eliminate the discrepancies in the compactedness of the shadow shape.
- (h) With the shape of the shadow finally generated, the area is intersected with the regions of the labeled region map in order to find the intersection part belonging to every different region. The intersected parts are evaluated according to its size to decide if they are large enough to be included in the output list.
- (h) The minimum bounding rectangular areas are calculated for each intersection part, and with these limits is possible to obtain the average tone of the intersection part from the reference image.
- (i) Finally the results are transferred back to the certainty shadow processing rule in the change detection system via mailbox. At the end of the program execution, the logical units used will be freed and an event flag is set to indicate a completion status.



i. **Exception handling**

Not used.

j. **Calling routines**

This program is initiated by the Prolog Interpreter during execution of the statement "'Simage' (shdpro.{},L)" in the "certainty-of-shadow" rule.

k. **Called routines**

JCPU:	Used to calculate the CPU time.
BLCKBRD:	To obtain connection with the blackboard storage.
REGFILL:	To fill a shadow projection representation.
QUADEQ:	To obtain the solution for a quadratic equation.
CONVERT:	To transform the ascii to integer values.
PLUMB:	Logical function used to verify if a point is inside or outside a given polygon.
FILENAM:	Used to provide a different filename for the progress report process while working in mode 1.

l. **Timing constraints**

Not observed.

m. **Other considerations**

- (a) In order to link this program, the following command will have to be given:  
Slink shdpro.blckbrd, plumb, quadeq, ufo\_create, ufo\_open filenam, jcpu, projmbr, projaff, regfill, blkbrd.opt.
- (b) The program assumes that sun light is the only light source, and that the background of a tested object is composed of one material. The certainty value of shadow is proportional to the estimated shadow predicted region over the average tone from the intersection part of the image with the shadow projection.
- (c) The changes in this program were related with the detection of some errors: the position in the vertex buffer of the projected and shadow coordinates was interchanged; also the use of the PREDREG array replaced properly the use of the MODEL array in some sentences.

5. **PROGRAM (FORTRAN) 'CRBKBD'**

a. **Functions performed**

The principal function of this program is the creation of the 'blackboard', which is an area defined for interprocess communication and the measurement of the

region's properties from a labeled picture, which will be stored in a common area called BOARD; the interested properties are: size, perimeter, compactedness, tone variations and minimum bounding rectangular.

**b. Input**

IN\_ARGS:           The number of arguments received through mailbox.  
LBLFILE:           The file that contains the labeled regions map.  
IMGFILE:           The file that contains the digitized image.

**c. Output**

The common area BOARD is generated during the blackboard generation process and also the same information is written in a region description file to be consulted by the Rule Base expert.

**d. Common area**

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

MODEL:            This matrix provides the 'model' 2-D projection.  
REGMAP:           Includes all the segmented regions.  
LR:                Side of the stereo pair of images.  
SUNANG:           Provides with the absolute sun direction.  
PLNCOEF:           Are the base plane coefficients A, B, C and D.  
FILESEQ:           A value used to identify the progress report files.  
COEF:             Coordinates corresponding to the exposure station.  
NREGS:            Is the number of regions identified in the photograph.  
AREA:             The region's area.  
PERI:             The region's perimeter.  
COMPACT:           Parameter that show a generated shape relation of the region.  
MTONE:            The average tone of each region.  
VTONE:            The tone variance of every region.  
MBR:             The minimum boundary rectangle for every region.  
DEBUG:            The type of debug process under execution (0,1 or 2).  
IMAGE:            The digitized image of the aerial photograph.

**e. Work area**

STRBUF:           A buffer to hold the input data for the program.  
PROFILE:           A file to hold the Prolog region description data.  
COMPACT:           Used to measure the compactedness of a region.

#### f. Parameters and modes

This program is designed to work with debug modes 0 and 1 (mode 2 is under development), when running in mode 1 there is additional information generated which will be displayed in the COMTAL environment to show the process progress report. The program uses as parameters, ISX and ISY with values of 128 which are the ranges in X and Y of the aerial photograph; MAXREG is 100 and it is the limit of regions to be obtained from the segmented image.

#### g. File handling

LBLFILE:	The file that contains the labeled regions map.
IMGFILE:	The file that contains the digitized image.
PROFILE:	A file to hold the Prolog region description data.
DEMON.DAT:	Output file that contains all information regarding the development of the change detection process.
PARAM.DAT:	Input file created by the Rule Base interpreter and provides with the input parameters to blackboard.
LDES.PRO:	Output file to be consulted by the Prolog Interpreter, it contains the region description information.

#### h. Program structure

The program for blackboard creation is summarized in the following steps:

- (a) The sun angle and plane coefficients arrays are initialized with the corresponding data from the sun direction and base plane equation. The data for left and right side of the stereo pair, and affine coefficients are also provided by the input parameters.
- (b) If debug mode 1 is in progress, then the original region map will be transferred to a demonstration file; REGMAP is also generated from the labeled file and the IMGFILE is assigned to the corresponding image file with the pixel values transformed to integers from 0 to 255; the field NREGS in the common area is also generated with the number of regions in the labeled file.
- (c) Using a set of different subroutines, the BOARD arrays for minimum bounding rectangular, size, perimeter, and tone are calculated and a region description file is generated.
- (d) The data to be fed through the blackboard for the different fortran subroutines, is also written in the Prolog region description file that later will be consulted by the Rule Base expert.

(e) At the end of the program execution, the logical units will be freed and an event flag is set to indicate a completion status.

i. Exception handling

Not used.

j. Calling routines

This program is initiated by the Prolog Interpreter during execution of the statement `“Simage’(crbkbd,{},X)”` in the initial stage of the Rule Base expert.

k. Called routines

BLKBRD: Serve as the data structure ‘blackboard’ creation routine.

CQLT1: Generates the MBR for the different regions.

PRMT2: Measures the perimeter of the specified regions.

AREA2: Measures the size of the labeled regions.

TONE: Get the mean tone and variance of the different regions.

STPT2: Measures the coordinates of the starting points on the border of a specified region.

FILENAM: Used to provide a different filename for the progress report process while working in mode 1.

l. Timing constraints

Not observed.

m. Other considerations

(a) In order to link this program, the following command will have to be given:  
`Slink crbkbd, blkbrd, convert, ufo_open, ufo_create, filenam, jcput,tone, sys$library:spiderlib/lib.`

(b) The process of assigning values to the variables in the ‘blackboard’ must be done after the file is opened, otherwise the values will be reset to zero because the memory pages assigned to the ‘blackboard’ are demand zero pages, and are initialized to 0 by the operating system.

(c) To calculate the compactedness of a region, the relation of area of the region against the area of a circle is used, as reference, the compactedness of a circle is 1.

## 6. PROGRAM (FORTRAN) 'PTPROJ'

### a. Functions performed

This program is used to obtain the reference point projected from 3-D to 2-D coordinates in order to establish a ratio between the predicted image conditions and the observed values obtained by the average of tone in the reference point for better accuracy the surrounding pixels to the reference point are also considered for the calculations.

### b. Input

INNARGS:        Number of input arguments.  
X, Y, Z:        Coordinates of the reference point.

### c. Output

IMGREF:        The reference point value calculated.  
N:              Number of Output arguments.

### d. Common area

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

MODEL:        This matrix provides the 'model' 2-D projection.  
REGMAP:       Includes all the segmented regions.  
LR:            Side of the stereo pair of images.  
SUNANG:       Provides with the absolute sun direction.  
PLNCOEF:      Are the base plane coefficients A, B, C and D.  
FILESEQ:      A value used to identify the progress report files.  
COEF:         Coordinates corresponding to the exposure station.  
NREGS:        Is the number of regions identified in the photograph.  
AREA:         The region's area.  
PERI:         The region's perimeter.  
COMPACT:      Parameter that show a generated shape relation of the region.  
MTONE:        The average tone of each region.  
VTONE:        The tone variance of every region.  
MBR:         The minimum boundary rectangle for every region.  
DEBUG:        The type of debug process under execution (0,1 or 2).  
IMAGE:        The digitized image of the aerial photograph.



**e. Work area**

UFO\_CREATE: Logical unit to be created.  
UFO\_OPEN: Logical unit to be opened.  
U, V, W: Real equivalent values of X, Y, Z.  
IC, JC: Coordinate in 2-D of the reference point.  
SFORDEF: Fortran attribute flags.  
SSECDEF: Attribute flags for private/global section creation and mapping.

**f. Parameters and modes**

Not used.

**g. File handling**

Not used.

**h. Program structure**

The program for reference point projection is summarized in the following steps:

- (a) The reference point projection program is initiated by the Prolog Interpreter, and the first task is to establish communication through the memory management interprocess routine: event flag association, mailbox communication and logical unit association.
- (b) The input parameters are loaded; the location of the reference point in 2-D is generated.
- (c) The average value of the surrounding points to the reference point included is calculated from the image; finally before returning, the logical units are freed and an event flag is set to indicate a completion status.

**i. Exception handling**

Not used.

**j. Calling routines**

This program is initiated by the Prolog Interpreter by using the command 'Simage' (project\_mbr, MBR, Proj\_MBR), in control rule No. 2.

**k. Called routines**

PROJECT: To project a 3-D coordinate to 2-D coordinate.  
XY2IJ: Perform the affine transformation.

- l. Timing constraints  
Not observed.
- m. Other considerations

## 7. SUBROUTINE (FORTRAN) 'AFFIJ'

### a. Functions performed

Perform the affine transformation (2-D image to 2-D image 6 parameter transformation) includes a rotation, non perpendicularity of the axes, two scale changes and two translations.

### b. Input

IX,IY: These are the Image coordinates to be affined.

### c. Output

IX,IY: Are the transformed 2-D image coordinates.

### d. Common area

Not used.

### e. Work area

PHOTO TO DIGITAL IMAGE TRANSFORMATION:

Transformation from the photograph coordinates to the digital image coordinates is called an affine transformation and represents a rotation, nonperpendicularity of the axes, two scale changes and two translations. This implies the solution of the equations:

$$XP = SX(\cos(BETA) - DELTA * \sin(BETA))XI - (SY * \sin(BETA))YI + C1$$

$$YP = SX(\sin(BETA) + DELTA * \cos(BETA))XI + (SY * \cos(BETA))YI + C2$$

where

$XP, YP$  = Photo coordinates

$XI, YI$  = Image coordinates.

$SX, SY$  = Scale changes.

$C1, C2$  = Translation in both axes.

$DELTA$  = Nonperpendicularity or affinity between the axes  $X, Y$ .

$BETA$  = Rotation angle.

$$A1 = SX(\cos(BETA) - DELTA * \sin(BETA))$$

$$A2 = SY(\sin(BETA) + DELTA * \cos(BETA))$$

$$B1 = -SY * \sin(BETA)$$

$$B2 = +SY * \cos(BETA)$$

This transformation is also called 6-parameter transformation. This process has been implemented in this subroutine AFFIJ.

f. **Parameters and modes**

g. **File handling**

Not used.

h. **Program structure**

Not used.

The program for affine transformation is summarized in the following steps: .

(a) Solve the affine transformation equations to convert the photo coordinates to display coordinates.

i. **Exception handling**

Not used.

j. **Calling routines**

SELREG: Select regions intersecting with the projected node.

SHADOW: Used to generate the shadow projections.

k. **Called routines**

Not used.

l. **Timing constraints**

Not observed.

m. **Other considerations**

Not used.

## 8. SUBROUTINE (FORTRAN) 'JCPUT'

a. **Functions performed**

Obtain the CPU time consumed in a specific program.

b. **Input**

XCPUT: Variable to hold the system CPU time.

c. **Output**

Not used.

d. **Common area**

Not used.

e. **Work area**

BUF: Buffer to hold the information required by SYSSGETJPI.

f. **Parameters and modes**

Not used

g. **File handling**

Not used

h. **Program structure**

The program for CPU time calculation is summarized in the following steps:

(a) **Generate the information necessary to call the system utility SYSSGETJPI.**

i. **Exception handling**

Not used.

j. **Calling routines**

SELREG: Select regions intersecting with the projected node.

SHDPRO: Shadow processing module.

REFINE: To refine the region's association with the node.

k. **Called routines**

Not used.

l. **Timing constraints**

Not observed.

m. **Other considerations**

Not used.

9. **SUBROUTINE (FORTRAN) 'PROJMBR'**

a. **Functions performed**

This program is used to project the 3-D MBR of the node projection into 2-D left and right image coordinates.

b. **Input**

INNARGS: Number of input arguments.

X, Y, Z: Coordinates of the MBR.

c. **Output**

MBR2D: The 2-D coordinates of the MBR.

d. **Common area**

Not used.

e. **Work area**

U, V, W: Real equivalent values of X, Y, Z.

IJTEMP: Hold temporarily the 2-D MBRs.

IOLUN: Logical unit to be used as input/output.

**f. Parameters and modes**

Not used.

**g. File handling**

DEMON.DAT:

Output file that contains all process messages.

**h. Program structure**

The program for minimum boundary rectangular calculation is summarized in the following steps:

- (a) The minimum boundary rectangular program is initiated by the Prolog Interpreter, and the first task is to establish communication through the memory management interprocess routine: event flag association, mailbox communication and logical unit association.

- (b) The input parameters are loaded; the location of MBR 2-D is generated.

- i. Exception handling**

- Not used.

- j. Calling routines**

- This program is initiated by the Prolog Interpreter by using the command 'Simage' (project\_mbr, MBR, Proj\_MBR) in the execution of control rule No. 2.

- k. Called routines**

- PROJECT: To project a 3-D coordinate to 2-D coordinate.

- XY2IJ: Perform the affine transformation.

- BLCKBRD: To obtain connection with the blackboard storage.

- l. Timing constraints**

- Not observed.

- m. Other considerations**

- Not used.

**10. SUBROUTINE (FORTRAN) 'REFINE'**

**a. Functions performed**

Refine the results provided by a basic segmentation algorithm according to the intersection part of the area mapped from a 'model' and the regions of a labeled image. The refine process implies the creation of new regions from the intersected parts and the reconfiguration of the whole region map in order to simplify the analysis of the change detection process.



**b. Input**

REGNO:           Region list associated with a specific node.  
NOREGS:          Number of regions listed for the chosen node.

**c. Output**

The common area BOARD is updated by including the new regions and its properties after the refine process. The elements modified are: REGMAP, AREA, PERI, COMPACT, MTONE and VTONE.

**d. Common area**

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

MODEL:           This matrix provides the 'model' 2-D projection.  
REGMAP:          Includes all the segmented regions.  
LR:               Side of the stereo pair of images.  
SUNANG:          Provides with the absolute sun direction.  
PLNCOEF:         Are the base plane coefficients A, B, C and D.  
FILESEQ:         A value used to identify the progress report files.  
COEF:            Coordinates corresponding to the exposure station.  
NREGS:           Is the number of regions identified in the photograph.  
AREA:            The region's area.  
PERI:            The region's perimeter.  
COMPACT:         Parameter that show a generated shape relation of the region.  
MTONE:           The average tone of each region.  
VTONE:           The tone variance of every region.  
MBR:             The minimum boundary rectangle for every region.  
DEBUG:           The type of debug process under execution (0,1 or 2).  
IMAGE:           The digitized image of the aerial photograph.

**e. Work area**

M\_AREA, M\_PERI, M\_MBR, M\_TONE, M\_VTONE, M\_COMPACT are the temporary variables assigned to hold the related information for the node in process as explained previously in part 4.

INTS\_AREA,       INTS\_PERI,       INTS\_TONE,       INTS\_VTONE,  
INTS\_COMPACT are the temporary variables assigned to hold the related information for the intersection part as explained previously in part 4.

NEW\_AREA, NEW\_PERI, NEW\_TONE, NEW\_VTONE, NEW\_COMPACT are the temporary variables assigned to hold the related information for the new created region as explained previously in part 4.

NOREGS:	The total number of regions identified in REGMAP.
RECT:	Temporary area to hold the MBR of the region under test.
FILESEQ:	A sequence to identify a demonstration file.
JP:	Array to hold a region without the intersection part.
IP:	Array to hold the intersection part.
RATIO1:	The intersection part over the node area ratio.
RATIO2:	The intersection part over the region size ratio.
REGLIST:	A list of regions neighboring another.
COUNTX,1:	Variables to hold the CPU time consumed by the program.
NEIGH_REGS:	The number of regions neighboring another.
BUF2D:	An area that shows the merged regions for debug purposes.
NCAND:	The candidate region with the most similar characteristics to the region in process.
NEW_NREGS:	The number of new regions obtained from the intersection part of a node.
NEW_LBL_IMAGE:	An area with new labels for all the new generated regions.

#### f. Parameters and modes

The following three variables must come from the Rule Base interpreter in the final version: LOW\_BOUND which is initialized to 5, is the size of candidate noise; IUP\_BOUND which is initialized to 30, is the maximum gray difference for small region merging; GRAY\_DIFF which is initialized to 10 is the maximum gray difference for big region merging. It also uses as a parameter, NEWREG, which is initialized to 20 and is the maximum number of new regions that could be generated after the splitting and merge process.

This program is designed to work with debug modes 0 and 1 (mode 2 is under development), when running in mode 1 there is additional information generated which will be displayed in the COMTAL environment to show the process progress report. The program uses as parameters, ISX and ISY with values of 128 which are the ranges in X and Y of the aerial photograph.

**g. File handling**

When working in mode 1, the files 'DEMFIL' and 'MSGFILE' will be opened by the subroutine FILENAM, and closed by the refine process after writing the debug information.

DEMON.DAT: Output file that contains all process messages.

**h. Program structure**

The following steps will be accomplished by the refine process:

- (a) Work area variables and parameters are initialized.
- (b) For every region that intersects the 'model', there are two new arrays generated: IP and JP which respectively contain the intersection part and the region without it; this will be used later when deciding if an intersection part is too small or not in relation to the node or region size.
- (c) The properties (area, perimeter, compactedness, gray differences) are measured for the 'model' and the intersection part; some debug information is sent to the demonstration files.
- (d) The size of the intersection parts is analyzed in relation to the size of the region and the 'model': if it is too big in relation to both, then this intersection part will not be considered for the splitting and merge process.
- (e) If the intersection is small, then it will be split according to the shape of the 'model', and the resulting regions will be merged with the neighbor region that have the most similar characteristics.
- (f) The new properties will be generated and the facts database will be updated properly.

**i. Exception handling**

Not used.

**j. Calling routines**

This subroutine is called by the Model guided split and merge program (MGSM) with the sentence: call refine (REGLIST, IN\_NARGS).

**k. Called routines**

JCPU:	Used to calculate the CPU time.
MEASREG:	Used to get the new properties of a group of regions.
NEIGHREGS:	It get all the neighbor regions from a selected one.
MEASONE:	Measure the area, tone, mbr and compactedness of the a selected region.
WRPROP:	Write properties of a region in a file to be consulted

by the Change Detection process.

**ASSREG:** Is used to generate new regions from a splitted intersection part of a node.

**FILENAM:** Used to provide a different filename for the progress report process while working in mode 1.

**l. Timing constraints**

Not observed.

**m. Other considerations**

In order to link this program, the following command will have to be given:

- (a) Slink mgsn, refine, assreg, neighreg, tone, measone, measreg, wrprop, jcpur, filenam, ufo\_create, ufo\_open, blkbrd, blkbrd/options, sysSlibrary:spider.lib
- (b) A noise region is defined as a regions whose size is smaller than LOW\_BOUND and its tone variance is greater than UP\_BOUND.
- (c) Changes to the program REFINER were made in the way that all the new data generated for the merged and splitted regions could be effectively transfered to the database and common area BOARD, so it could be used appropriately by the incoming processes. Other changes performed to the program are related with the insertion of the timing routines.

## **11. SUBROUTINE (FORTRAN) 'BLKBRD'**

**a. Functions performed**

The principal function of this program is to create and map a section in memory to serve as a data structure for the 'blackboard' interprocess communication area.

**b. Input**

Not used.

**c. Output**

Not used.

**d. Common area**

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

**MODEL:** This matrix provides the 'model' 2-D projection.

**REGMAP:** Includes all the segmented regions.

**LR:** Side of the stereo pair of images.

**SUNANG:** Provides with the absolute sun direction.

PLNCOEF:	Are the base plane coefficients A, B, C and D.
FILESEQ:	A value used to identify the progress report files.
COEF:	Coordinates corresponding to the exposure station.
NREGS:	Is the number of regions identified in the photograph.
AREA:	The region's area.
PERI:	The region's perimeter.
COMPACT:	Parameter that show a generated shape relation of the region.
MTONE:	The average tone of each region.
VTONE:	The tone variance of every region.
MBR:	The minimum boundary rectangle for every region.
DEBUG:	The type of debug process under execution (0,1 or 2).
IMAGE:	The digitized image of the aerial photograph.

**e. Work area**

UFO_CREATE:	Logical unit to be created.
UFO_OPEN:	Logical unit to be opened.
ADDR:	To provide the limits of the background area.
SFORDEF:	Fortran attribute flags.
SSECDEF:	Attribute flags for private, global section creation and mapping.

**f. Parameters and modes**

Not used.

**g. File handling**

Not used.

**h. Program structure**

The program for blackboard creation is summarized in the following steps:

- (a) The blackboard creation program is initiated by the Prolog Interpreter, and the first task is to establish communication through the memory management interprocess routine: event flag association, mailbox communication and logical unit association. Also the input parameters are loaded from the file PARAM.DAT and the labeled and image files are opened.
- (b) This subroutine uses the SYSSCRMPSC system utility to create and map a section in the memory, it is the actual blackboard area the one being created and mapped.



i. **Exception handling**

Not used.

j. **Calling routines**

This subroutine is called by CRBKBD and all the fortran programs that want to get the interprocess communication established; they are:

MGSIM:	Main program of the 'model' guided split and merge process.
SELREG:	Select regions intersecting with the projected node.
SHDPRO:	Shadow processing module.
PTPROJ:	Used to generate a point projection (3-D to 2-D).

k. **Called routines**

The system utility LIBSGET\_LUN allocates one logical unit from a system process wide pool.

l. **Timing constraints**

Not observed.

m. **Other considerations**

(a) Slink crbkbd, blkbrd, convert, ufo\_open, ufo\_create, filenam, jcput.tone, sysSlibrary:spiderlib/lib.

## 12. SUBROUTINE (FORTRAN) 'PROJECT'

a. **Functions performed**

The main function of this subroutine is to perform a 3-D earth coordinate to 2-D image coordinate transformation for a given point coordinates.

b. **Input**

X, Y, Z: These are the Object coordinates.

c. **Output**

XIMA, YIMA: Are the transformed 2-D image coordinates.

d. **Common area**

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

MODEL:	This matrix provides the 'model' 2-D projection.
REGMAP:	Includes all the segmented regions.
LR:	Side of the stereo pair of images.
SUNANG:	Provides with the absolute sun direction.
PLNCOEF:	Are the base plane coefficients A, B, C and D.
FILESEQ:	A value used to identify the progress report files.

COEF:	Coordinates corresponding to the exposure station.
NREGS:	Is the number of regions identified in the photograph.
AREA:	The region's area.
PERI:	The region's perimeter.
COMPACT:	Parameter that show a generated shape relation of the region.
MTONE:	The average tone of each region.
VTONE:	The tone variance of every region.
MBR:	The minimum boundary rectangle for every region.
DEBUG:	The type of debug process under execution (0,1 or 2).
IMAGE:	The digitized image of the aerial photograph.

**e. Work area**

NCHARS:	The number of characters in the node name.
FOCUS:	Focal length.
X0, Y0:	Fiducial Center.
DENOM:	Used for simplifying the transformation (denominator).
OMEGA:	Exterior orientation parameter: rotation about the X axis.
PHI:	Exterior orientation parameter: rotation about the Y axis.
KAPPA:	Exterior orientation parameter: rotation about the Z axis.
XL, YL, ZL:	Exposure station: Object coordinates of the perspective center.

COSOME, SINOME, COSPHI, SINPHI, COSKAP, SINKAP are respectively the cosine and sine of the angles omega, phi and kappa, used as exterior orientation parameters; the angle units is in radians.

M11 to M33 are the Orientation Matrix coefficients.

**f. Parameters and modes**

The interior orientation parameters are: X0, Y0 and FOCUS; the exterior orientation parameters are: XL, YL, ZL, OMEGA, PHI, KAPPA.

This program is designed to work with debug modes 0 and 1 (mode 2 is under development), when running in mode 1 there is additional information generated which will be displayed in the COMTAL environment to show the process progress report.

**g. File handling**

DEMON.DAT: Output file that contains all process messages.

#### **h. Program structure**

The program for data projection is summarized in the following steps:  
following steps:

- (a) The Orientation Matrix is generated for every side of the stereo pair image, and the image coordinate system is constructed by using the orientation matrix coefficients, the interior orientation parameters and the exterior orientation parameters.

- i. **Exception handling**

- Not used.

- j. **Calling routines**

- PROJMBR: Project a MBR from 3-D to 2-D coordinates.

- SELREG: Select regions intersecting with the projected node.

- SHADOW: Used to generate the shadow projections.

- PTPROJ: Used to generate a point projection (3-D to 2-D).

- k. **Called routines**

- Not used.

- l. **Timing constraints**

- Not observed.

- m. **Other considerations**

- Not used.

### **13. SUBROUTINE (FORTRAN) 'XY2IJ'**

- a. **Functions performed**

Perform the affine transformation (2-D photograph to 2-D image 6 parameter transformation) includes a rotation, non perpendicularity of the axes, two scale changes and two translations.

- b. **Input**

- X, Y: These are the Photograph coordinates.

- c. **Output**

- I, J: Are the transformed 2-D image coordinates.

- d. **Common area**

- Not used.

e. **Work area**

PHOTO TO DIGITAL IMAGE TRANSFORMATION:

Transformation from the photograph coordinates to the digital image coordinates is called an affine transformation and represents a rotation, nonperpendicularity of the axes, two scale changes and two translations. This implies the solution of the equations:

$$XP = SX(\cos(BETA) - DELTA * \sin(BETA))XI - (SY * \sin(BETA))YI + C1$$

$$YP = SX(\sin(BETA) + DELTA * \cos(BETA))XI + (SY * \cos(BETA))YI + C2$$

where

$XP, YP$  = Photo coordinates

$XI, YI$  = Image coordinates.

$SX, SY$  = Scale changes.

$C1, C2$  = Translation in both axes.

$DELTA$  = Nonperpendicularity or affinity between the axes  $X, Y$ .

$BETA$  = Rotation angle.

This transformation is also called 6-parameter transformation. This process has been implemented in this subroutine XY2IJ. The following data has been provided for the analysis of the aerial photographs under test:

$$A1 = SX(\cos(BETA) - DELTA * \sin(BETA))$$

$$A2 = SY(\sin(BETA) + DELTA * \cos(BETA))$$

$$B1 = -SY * \sin(BETA)$$

$$B2 = +SY * \cos(BETA)$$

f. **Parameters and modes**

g. **File handling**

Not used.

h. **Program structure**

Not used.

The program for affine transformation is summarized in the following steps:

- (a) Solve the affine transformation equations to convert the photo coordinates to display coordinates.

i. **Exception handling**

Not used.

j. **Calling routines**

PROJMBR: Project a MBR from 3-D to 2-D coordinates.  
SELREG: Select regions intersecting with the projected node.  
SHADOW: Used to generate the shadow projections.  
PTPROJ: Used to generate a point projection (3-D to 2-D).

k. **Called routines**

Not used.

l. **Timing constraints**

Not observed.

m. **Other considerations**

Not used.

#### 14. **SUBROUTINE (FORTRAN) 'CONVERT'**

a. **Functions performed**

Convert a string buffer from the character data type to integer data type.

b. **Input**

STRBUF: The character string containing the input data.  
NUM: The number of values to be transferred back.

c. **Output**

LIST: The list of integer values to be returned.

d. **Common area**

The common area called BOARD has been developed to provide interprocess communication through a background storage and includes:

MODEL: This matrix provides the 'model' 2-D projection.  
REGMAP: Includes all the segmented regions.  
LR: Side of the stereo pair of images.  
SUNANG: Provides with the absolute sun direction.  
PLNCOEF: Are the base plane coefficients A, B, C and D.  
FILESEQ: A value used to identify the progress report files.  
COEF: Coordinates corresponding to the exposure station.  
NREGS: Is the number of regions identified in the photograph.  
AREA: The region's area.  
PERI: The region's perimeter.  
COMPACT: Parameter that show a generated shape relation of the region.  
MTONE: The average tone of each region.



VTONE:	The tone variance of every region.
MBR:	The minimum boundary rectangle for every region.
DEBUG:	The type of debug process under execution (0,1 or 2).
IMAGE:	The digitized image of the aerial photograph.

e. **Work area**

ISIGN:	The sign of the output value.
NO:	Number of elements that are being generated.

f. **Parameters and modes**

The variable ISIGN has the values '1' for positive and '-1' for negative values; the character buffer being transformed is 72 characters long.

g. **File handling**

DEMON.DAT: Output file that contains all process messages.

h. **Program structure**

The following steps will be accomplished by the conversion process:

- (a) For every character in the string buffer, check first the sign of the number, then verify that the number is an ascii code between '0' and '9'. check different numbers by detecting the ',' code and finish when locating the closing bracket of the list of characters from the input buffer.

i. **Exception handling**

This program will issue error messages when detecting incompatible number of integers.

j. **Calling routines**

This program is called by issuing the command ' call convert (strbuf, list, inum) ' and the following routines are calling this program:

CRBKBD:	Generates the blackboard area.
SHADOW:	Used to generate the shadow projections.
SELREG:	Select regions intersecting with the projected node.
SHDPRO:	Shadow processing module.

k. **Called routines**

l. **Timing constraints**

Not observed.

m. **Other considerations**

Not used.

## 15. SUBROUTINE (FORTRAN) 'SHADOW'

### a. Functions performed

The function of this program is to find the 2-D shadow coordinate projected from a 3-D coordinate object.

### b. Input

X, Y, Z:           The coordinates of the object.  
SUNANG:           Array that gives the sun direction.  
PLNCOEF:           Values for base plane coefficients A, B, C and D.  
COEF:             Affine coefficients.  
LR:                Side of the stereo pair of images.

### c. Output

IS, JS:            The shadow image coordinates.

### d. Work area

SX,SY,SZ:          Are the projected shadow coordinates in 3-D.  
R:                 The distance from the object to the ground.

### e. Parameters and modes

This program is designed to work with debug modes 0 and 1 (mode 2 is under development), when running in mode 1 there is additional information generated which will be displayed in the COMTAL environment to show the process progress report. The program uses as parameters, ISX and ISY with values of 128 which are the ranges in X and Y of the aerial photograph.

### f. File handling

When working in mode 1, the files 'DEMFILE' and 'MSGFILE' will be opened by the subroutine FILENAM, and closed by the refine process after dumping the debug information.

DEMON.DAT:        Output file that contains all process messages.

### g. Program structure

The following steps will be accomplished by the shadow process:

- (a) This program uses the plane coefficients of the building background base in order to obtain the distance from the vertex to the ground ; with this information, an the sun position, the estimated ground coordinates where the shadow will be located are obtained.
- (b) Then this coordinates of the shadow in 3-D are transformed in 2-D.

**h. Exception handling**

used.

**i. Calling routines**

This subroutine is called by SELREG using the command ' call shadow(x,y,z, sunang,plncoef,lr,is,js).

**j. Called routines**

PROJECT: To project a 3-D coordinate to 2-D coordinate.

AFFIJ: Generate additional affine transformation to the image.

XY2IJ: Perform the affine transformation.

FILENAM: Used to provide a different filename for the progress report process while working in mode 1.

**k. Timing constraints**

Not observed.

**l. Other considerations**

Not used.

**16. SUBROUTINE (FORTRAN) 'FILENAM'**

**a. Functions performed**

This subroutine generates a sequence of different demonstration files included under the variables DEMFILE and MSGFILE while running the Change Detection System under debug mode 1.

**b. Input**

FILESEQ: Value that shows the actual order for file creation.

**c. Output**

FILENAM: Character string buffer to be used for opening the Demonstration and Message files.

**d. Common area**

Not used.

**e. Work area**

Not used.

**f. Parameters and modes**

Not used.

**g. File handling**

The demonstration files with sequence DEM001 to DEM999 and the message files with sequence MSG001 to MSG999 are created and opened in the logical units 1 (DEMFILE) and 11 (MSGFILE) according to the Change Detection process needs.

**h. Program structure**

The program for file names generation is described as follows:

- (a) Ascii characters are generated by translating the integer sequence FILESEQ digit by digit, and incrementing the sequence previously for every new file needed.

**i. Exception handling**

Not used.

**j. Calling routines**

This program is called when running under mode debug '1' by the following programs:

SELREG:	Select regions intersecting with the projected node.
REFINE:	To refine the region's association with the node.
SHDPRO:	Shadow processing module.
CRBKBD:	Generates the blackboard area.
SHADOW:	Used to generate the shadow projections.
MEASONE:	Measure the area, tone, nibr and compactedness.
MEASREG:	Used to get the new properties of a group of regions.

**k. Called routines**

Not used.

**l. Timing constraints**

Not observed.

**m. Other considerations**

Not used

**17. SUBROUTINE (FORTRAN) 'QUADEQ'**

**a. Functions performed**

This program finds the solution of a quadratic equation.

**b. Input**

A, B, C:                These are the equation coefficients.

- c. **Output**  
X1, X2:                These are the calculated real solutions.
- d. **Common area**  
Not used.
- e. **Work area**  
Not used.  
DELTA:                The expression  $B^2 - 4AC$
- f. **Parameters and modes**  
Not used.
- g. **File handling**  
DEMON.DAT:        Output file that contains all process messages.
- h. **Program structure**  
(a) Uses the formula  $\text{solution} = (-B + \sqrt{\text{delta}})/(2A)$  to find the real roots of the equation.
- i. **Exception handling**  
This subroutine checks for no real solutions or input coefficient errors.
- j. **Calling routines**  
This subroutine is called by the following programs:  
SHDPRO:                Shadow processing module.
- k. **Called routines**  
Not used.
- l. **Timing constraints**  
Not observed.
- m. **Other considerations**  
Not used.

## 18. SUBROUTINE (FORTRAN) 'MEASREG'

- a. **Functions performed**  
The principal function of this program is to measure the properties of a labeled picture, the interested properties are: size, average gray level, perimeter, compactedness; this data is obtained for a group of regions.
- b. **Input**  
IP:                      The intersected part.  
IMAGE:                The file that contains the image picture.  
ISX, ISY:             The image dimensions in X and Y (use 128).



NREGS:	Number of regions in the region map.
IRN:	The index region number (selected region for the measurements.
MAXREG:	Value used for dynamic dimension of the labeled region buffer.

**c. Output**

AREA:	The region's area.
PERI:	The region's perimeter.
COMPACT:	The region's compactedness.
MTONE:	The average tone of the region's gray values.
VTONE:	The region's tone variance.

**d. Common area**

Not used.

**e. Work area**

Not used.

**f. Parameters and modes**

Not used.

**g. Program structure**

The program for region measurement is summarized in the following steps:

- (a) Using a set of different subroutines, the different region properties are calculated for the specified regions.

**h. Exception handling**

Not used.

**i. Calling routines**

This subroutine is called by REFINE, using the command: call measreg(ip, image, isx, isy, nregs, irn, area, peri, compact, mbr, mtone, vtone, maxreg).

**j. Called routines**

CQLT1:	Generates the MBR for the different regions.
PRMT2:	Measures the perimeter of the specified regions.
AREA2:	Measures the size of the labeled regions.
TONE:	Get the mean tone and variance of the different regions.
STPT2:	Measures the coordinates of the starting points on the border of a specified region.
FILENAM:	Used to provide a different filename for the progress report process while working in mode 1.

k. **Timing constraints**

Not observed.

l. **Other considerations**

- (a) To calculate the compactedness of a region, the relation of area of the region against the area of a circle is used, as reference, the compactedness of a circle is 1.

19. **SUBROUTINE (FORTRAN) 'TONE'**

a. **Functions performed**

Find the mean and variance of the gray level for a specified region.

b. **Input**

LABEL: The labeled region file.

IMAGE: The image file.

ISX, ISY: The image dimensions (assigned a value of 128).

REGNO: The region number being processed.

RECT: The Minimum Boundary Rectangular of the region.

c. **Output**

MEAN: Average gray value of the region.

VARIANCE: Variance of the gray level in the region.

d. **Common area**

Not used.

e. **Work area**

NOPOINTS: Number of pixels integrating the region.

SUM: Sum of the image values.

SUMSQ: Sum of the image values squared.

f. **Parameters and modes**

Not used.

g. **File handling**

DEMON.DAT: Output file that contains all process messages.

h. **Program structure**

The program for tone generation is summarized in the following steps:

- (a) Obtain the mean =  $\text{SUM} / \text{NOPOINTS}$ .
- (b) Obtain  $\text{VARIANCE} = ((\text{SUMSQ} - \text{SUM} * \text{SUM}) / \text{NOPOINTS}) / (\text{NOPOINTS} - 1)$ .

i. **Exception handling**

Not used.

j. **Calling routines**

The following routines call this program using the command `call tone(label, image, isx, isy, mean, variance, RegNo, rect)`.

CRBKBD: Generates the blackboard area.

MEASONE: Measure the area, tone, mbr and compactedness.

MEASREG: Used to get the new properties of a group of regions.

k. **Called routines**

Not used.

l. **Timing constraints**

Not observed.

m. **Other considerations**

Not used.

## 20. SUBROUTINE (FORTRAN) 'WRPROP'

a. **Functions performed**

To write the new Region Properties into the file NEWPROP.DAT that will be consulted by the Prolog Interpreter.

b. **Input**

I: The region number.

AREA: The region's area.

PERI: The region's perimeter.

COMPACT: The region's compactedness.

MTONE: The average tone of the region's gray values.

VTONE: The region's tone variance.

c. **Output**

Not used.

d. **Common area**

Not used.

e. **Work area**

Not used.

f. **Parameters and modes**

Not used.

g. **File handling**

The file NEWPROP.DAT is output file opened in logical unit No. 13.

h. **Program structure**

The program for tone generation is summarized in the following steps:

- (a) The input parameters are formatted and written to be loaded as new facts in the file NEWPROP.DAT with the following mnemonics preceding them: REG\_SIZE, REG\_PERI, REG\_COMPACT, REG\_TONE.

i. **Exception handling**

Not used.

j. **Calling routines**

The following programs call this subroutine:

REFINE: To refine the region's association with the node.

The command used is: call wrprop (i, area, peri, compact, mtone, vtone)

k. **Called routines**

Not used.

l. **Timing constraints**

Not observed.

m. **Other considerations**

At present time the facts for perimeter and compactedness are not being used.

**21. SUBROUTINE (FORTRAN) 'MEASONE'**

a. **Functions performed**

The principal function of this program is to measure the properties of a labeled picture, the interested properties are: size, average gray level, perimeter, compactedness; this data is obtained for a particular region.

b. **Input**

IP: The intersected part.

IMAGE: The file that contains the image picture.

ISX, ISY: The image dimensions in X and Y (use 128).

NREGS: Number of regions in the region map.

IRN: The index region number (selected region for the measurements.

MAXREG: Value used for dynamic dimension of the labeled

region buffer.

**c. Output**

AREA: The region's area.  
PERI: The region's perimeter.  
COMPACT: The region's compactedness.  
MTONE: The average tone of the region's gray values.  
VTONE: The region's tone variance.

**d. Common area**

Not used.

**e. Work area**

Not used.

**f. Parameters and modes**

Not used.

**g. Program structure**

The program for region measurement is summarized in the following steps:

- (a) Using a set of different subroutines, the different region properties are calculated for the specified region.

**h. Exception handling**

Not used.

**i. Calling routines**

This subroutine is called by REFINE, using the command: call measone (ip, image, isx, isy, nregs, irn, area, peri, compact, mbr, mtone, vtone, maxreg).

**j. Called routines**

CQLT1: Generates the MBR for the different regions.  
PRMT2: Measures the perimeter of the specified regions.  
AREA2: Measures the size of the labeled regions.  
TONE: Get the mean tone and variance of the different regions.  
STPT2: Measures the coordinates of the starting points on the border of a specified region.  
FILENAM: Used to provide a different filename for the progress report process while working in mode 1.

**k. Timing constraints**

Not observed.



1. **Other considerations**

- (a) To calculate the compactedness of a region, the relation of area of the region against the area of a circle is used, as reference, the compactedness of a circle is 1.

**22. SUBROUTINE (FORTRAN) 'NEIGHREGS'**

a. **Functions performed**

Generate a list of region neighbors to a specified one.

b. **Input**

ISX, ISY: The image coordinate dimensions (value of 128).

LBL\_IMG: Is the labeled region map.

RECT: Limit of search enclosing the selected region.

REGNO: The mother region from where new smaller regions were generated.

LABEL: The selected region for which neighbors regions are search.

c. **Output**

REGLIST: List of regions neighboring a specified one.

NREGS: Number of regions neighboring a specified one.

NEW\_LBL\_IMG: The labeled map after splitting an intersection part into new several smaller regions.

d. **Common area**

Not used.

e. **Work area**

JX, JY: Variables to hold the location of search.

DISPL: Are the displacements in the four possible directions of searching (up, down, left, right).

f. **Parameters and modes**

Not used.

g. **File handling**

DEMON.DAT: Output file that contains all process messages.

h. **Program structure**

The program for neighbor location is summarized in the following steps:

- (a) Get the input parameters and use the MBR of the region intersecting the 'model' to limit the search for the regions not being equal to the mother region, and that are surrounding the specified new region.
- (b) Search in all directions to find the neighbor regions.
  - i. **Exception handling**  
Not used.
  - j. **Calling routines**  
This subroutine is called by REFINE, using the command: call neighregs (new\_lbl\_img, lbl\_img, isx, isy, label, rect, nregs, RegList, RegNo ).
  - k. **Called routines**  
Not used.
  - l. **Timing constraints**  
Not observed.
  - m. **Other considerations**

## 23. SUBROUTINE (FORTRAN) 'ASSREG'

### a. Functions performed

Assign new region labels to the regions formed by extracting an intersection part from an old region.

### b. Input

LABEL:	Original region number being clipped off.
RECT:	MBR enclosing the region being clipped off.
ISX, ISY:	The coordinates sizes of the image.
LBL_IMG:	Intersection area to be relabeled.

### c. Output

NEW_NREGS:	The number of new regions being generated.
REGION:	The new labeled intersection area.

### d. Common area

Not used.

### e. Work area

TABLE:	Array with character values ('1' to '10') to relabel the new created regions.
KX, KY:	Stack locations for the coordinates of the region being analyzed.
STATUS:	Logical values to mark the top address of the stack;

if top = 0, then status = .FALSE..

f. **Parameters and modes**

Not used.

g. **File handling**

FOR008.DAT: Output file used for debug purposes.

h. **Program structure**

The following steps will be accomplished by the assign region process:

- (a) Reset the new labeled area to zero value.
- (b) Identify the intersection region to be reassigned and generate the new labeled image using different label values for the isolated areas.
- (c) To identify isolated parts in the splited image, ASSREG calls in a recursive method to the subroutine GENTEST, which test all the boundary pixels to the one being analyzed, and by this procedure establishes individuality.
- (d) the results are transfered back via output parameters, and Fortran file No. 8 is used for debugging purposes; the MBR being analyzed will be dumped as well as the new labeled region buffer.

i. **Exception handling**

This program issue error messages when there exist an stack overflow situation.

j. **Calling routines**

This subroutine is called by the REFINE process with the command: call assreg( lbl\_img, region, isx, isy, label, rect, new\_nregs).

k. **Called routines**

GEN\_TEST: Test new created regions for individuality.

POP, PUSH: Used to push and pop values from a local stack.

l. **Timing constraints**

Not observed.

m. **Other considerations**

None.

## 24. SUBROUTINE (FORTRAN) 'REFLEC'

a. **Functions performed**

To modify the calculated tone assigned to every node as a function of the sun rays incident angle over the node's surface.

**b. Input**

ROOF:	Are the node 3-D vertices coordinates.
PLNCOEF:	Are the base plane coefficients.
SUNANG:	Values that show the sun rays direction.
NODENAME:	Name of the node being analyzed.
LR:	Side of the stereo pair of images.
NCHARS:	Number of characters in the node's name.
EXPX,Y,Z:	Object coordinates for the perspective center.

**c. Output**

The calculated new tone is transfered back via mailbox.

**d. Common area**

Not used.

**e. Work area**

ALBEDO:	Reflectance factor.
E0:	Source irradiance.
ES:	Radiance of a hemispherical uniform source.
ANGTOT:	Acumulative Node normal azimuth.
ELETOT:	Acumulative Node normal elevation.
SANGLE:	Sun azimuth.
SELEVA:	Sun elevation.
BANGLE:	Base normal azimuth.
BELEVA:	Base normal elevation.
ROFANG:	Node normal azimuth.
ROFELE:	Node normal elevation.
OBSANG:	Observer azimuth.
OBSELE:	Observer elevation.
SGRADP,Q:	Sun gradient coefficients P and Q.
RGRADP,Q:	Node gradient coefficients P and Q.
TEMP:	Temporary array used to obtain the average plane equations.
A11 to C22:	Matrix coefficients used to get plane's normal coefficients.
A, B, C:	Plane equation coefficients.
SUNNOR:	Incident angle: angle between sun direction and normal to the plane formed by the node vertices.
EMERGE:	Emergent angle: angle between observer direction and normal to the plane formed by the node vertices.

**f. Parameters and modes**

Not used.

**g. File handling**

Not used.

**h. Program structure**

The following steps will be accomplished by the reflection process:

- (a) Work area variables are initialized; the sun angle and elevation are obtained from the sun angle direction coefficients and modified with respect to the base plane reference; the sun gradient is obtained.
- (b) Get the plane coefficients for the different vertex combination, to minimize the average error of the plane formed by the node's vertices. Generate the matrix to calculate the coefficients and solve the coefficient equation; once solved, obtain the roof's angle and elevation normal to the plane; get the gradient of the node's normal.
- (c) With the sun data and the normal data, calculate the angle between the sun and the normal of the node's roof, if it is less than 90 degrees, it means that the surface is being illuminated directly by the sun rays; get the observer direction, angle and elevation and its gradient to solve the emergent angle between the observer position and the normal to the node's surface normal.
- (d) With the incident and emergent angle, calculate the tone of the surface according to the position of the node's surface normal including the different effect caused by the incident and emergent angles.
- (e) Transfer the parameters back to the calling program which will be send to the Prolog Interpreter via mailbox.

**i. Exception handling**

This program checks for error codes generated by RSAM; check if the region number exceeds 200, which is the maximum value; provide with error messages.

**j. Calling routines**

This program is called by SELREG with the command call reflec( roof, tone, plncoef, sunang, nodename, lr, nchars).

**k. Called routines**

ANGEL: Used to calculate the normal's azimuth and elevation.

DIRECT: To obtain direction from the sun and from observer.



- l. **Timing constraints**  
Not observed.
- m. **Other considerations**  
None.

## 25. FUNCTION (FORTRAN) 'PLUMB'

### a. Functions performed

This logical function test whether a point is inside or outside a polygon.

### b. Input

X0, Y0:               The test point coordinates.  
POLYGON:             Array of vertices forming the polygon.  
NEDGES:              Number of edges in the polygon.

### c. Output

The logical values .TRUE. or .FALSE. are generated for the variable PLUMB corresponding to the situation if the point is inside of the polygon.

### d. Common area

Not used.

### e. Work area

NINTSEC:             If the number of intersections is odd, generate  
a logical value of .TRUE. otherwise .FALSE..

### f. Parameters and modes

Not used.

### g. File handling

DEMON.DAT:   Output file that contains all process messages.

### h. Program structure

The program is summarized in the following steps:

- (a) This program uses pure analytic geometry to find if the point lies in the same side of all the lines joining the vertices of the polygon.

### i. Exception handling

This program gives warning messages if the point being analyzed, lies on the polygon itself, or if the plumb line runs through a line joining two adjacent vertices of the polygon.

j. **Calling routines**

This program is called by SHDPRO using the statement call plumb( X0, Y0, POLYGON, NEDGES).

k. **Called routines**

Not used.

l. **Timing constraints**

Not observed.

m. **Other considerations**

Not used

**26. SUBROUTINE (PASCAL) 'REGFILL'**

a. **Functions performed**

Filling the interior of polygons.

b. **Input**

A polygon which is represented by a sequence of coordinate pairs with the following form  $\{(X_1, Y_1), (X_2, Y_2), (X_3, Y_3), \dots, (X_n, Y_n)\}$ , where

(1)  $(X_1, Y_1) = (X_n, Y_n)$  and

(2)  $1 \leq X_i, Y_i \leq (\text{MAXCOL or MAXROW})$ .

c. **Output**

A labeled image where the interior of a polygon is represented by "1"

REFERENCE: see ( Ref. 20 ).

d. **Common area**

Not used.

e. **Program structure**

The following steps will be accomplished by the region filling process:

Insert a node in a list in order (Procedure InsertNode).

Begin the Main Program.

Open output labeled file.

Clear edge table.

Create a Bucket-sorted Edge Table.

1. Set y to the smallest y-coordinate which has an entry in the ET.

2. Initialize the AET (Active Edge Table) to be empty.

3. Repeat until the AET and ET are empty.

3.1 Merge information from ET bucket y with the AET, maintaining  
AET sort order on x.

3.2 Fill in desired pixel values on scan line  $y$  by using pairs of  $x$  coordinate from the AET.

3.3 Remove from the AET those entries for which  $y = \max$ .

3.4 For each entry remaining, in AET, replace  $x$  by  $x + 1/m$ .

This places the next scan line intersection into each entry in the AET.

3.5 Resort the AET based on  $X$ -min.

3.6 Increment  $y$  by 1.

While step 3. Print out the labeled picture.

End.

f. **Calling Routines:**

SELREG: Select regions intersecting with the projected node.

SHDPRO: Shadow processing module.

g. **Exception handling**

Not used.

h. **Timing constraints**

Not observed.

i. **Other considerations**

None.

# APPENDIX C

## PROGRAM REFLEC

```

C
C
C October 15, 1986
C
C
C      subroutine reflec(roof,tone,plncoef,sunang,nodename,lr,
*                               nchars)
C
C      implicit real*8(a-h,o-z),integer(i-n)
C      dimension roof(4,3),plncoef(4),sunang(3),temp(3,3)
C      dimension expx(2),expy(2),expz(2)
C      integer tone
C      character*8 nodename
C
C      data expx(1),expx(2)/-2693765.9,-2695594.2/
C      data expy(1),expy(2)/-4304520.4,-4303718.9/
C      data expz(1),expz(2)/+3859018.3,+3858623.0/
C
C      work area initialization
C
C      pi=4.0*datan(one)
C      albedo=1.0
C      source irradiance e0
C      e0=153.5
C      uniform hemispherical source irradiance es
C      es=58.82
C      angtot=0.0
C      eletot=0.0
C
C      obtain the sun angle and elevation
C
C      call direct(sangle,seleva,sunang(1),sunang(2),sunang(3))
C
C      get the sun direction with reference to the base plane
C
C      call angel(bangle,beleva,plncoef(1),plncoef(2),plncoef(3))
C      sangle=sangle+bangle
C      seleva=seleva+beleva
C      if(sangle.gt.(2.*pi))sangle=sangle-(2.*pi)
C
C      calculation of the sun gradient
C
C      sgradp=-1.0*dsin(sangle)*(1./dtan(seleva))
C      sgradq=-1.0*dcos(sangle)*(1./dtan(seleva))
C
C      get all the plane coefficients for different roof
C      vertex combinations.
C
C      do l = 1,4
C        do j = 1,3
C          do k = 1,3
C            jj=j+l-1
C            if(jj.gt.4)jj=jj-4
C            temp(j,k)=roof(jj,k)
C          enddo k
C        enddo j
C
C      get matrix calculation parameters
C
C      all=temp(2,2)-temp(1,2)

```

```

a12=temp(2,3)-temp(1,3)
a21=temp(3,2)-temp(1,2)
a22=temp(3,3)-temp(1,3)
b12=temp(2,1)-temp(1,1)
b22=temp(3,1)-temp(1,1)
b11=a12
b21=a22
c11=b12
c21=b22
c12=a11
c22=a21

C
C get the coefficients of the roof under consideration
C
a=a11*a22-a21*a12
b=b11*b22-b21*b12
c=c11*c22-c21*c12

C
C get the angle and elevation of the roof
C
call angel(rofang,rofele,a,b,c)
angtot=angtot+rofang
eletot=eletot+rofele

C
enddo | 1

C
C average normal to roof angle and elevation
C
angtot=angtot/4.
eletot=eletot/4.

C
C gradient of the roof
C
rgradp=-1.0*dsin(angtot)*(1./dtan(eletot))
rgradq=-1.0*dcos(angtot)*(1./dtan(eletot))

C
C calculation of the angle incident I between the
C sun and the normal to the roof
C
dnum=1.0+sgradp*rgradp+sgradq*rgradq
deno1=dsqrt(1.0+sgradp**2+sgradq**2)
deno2=dsqrt(1.0+rgradp**2+rgradq**2)
sunnor=dacos(dnum/(deno1*deno2))

C
C if the incident angle is greater than 90 degrees, then
C the node is not directly illuminated by the sun rays
C
if (sunnor.gt.(pi/2.0))return

C
C calculation of the emergent angle E between the
C exposure station and the normal to the roof
C
call direct(obsang,obsele,expz(lr),expy(lr),expz(lr))

C
C get observer gradient
C
ogradp=-1.0*dsin(obsang)*(1./dtan(obsele))
ogradq=-1.0*dcos(obsang)*(1./dtan(obsele))

C
C get the angle E
C
dnum=1.0+ogradp*rgradp+ogradq*rgradq
deno1=dsqrt(1.0+ogradp**2+ogradq**2)
deno2=dsqrt(1.0+rgradp**2+rgradq**2)
emerge=dacos(dnum/(deno1*deno2))

C
C calculation of the new tone for the node

```



```

c
c
c
c
c      calculating the new tone for the roof
c
c      smc=dble(tone)+(e0/pi)*albedo
c      c*cos(sunnor)+es*albedo*((1.+dcos(emerger))/2.)
c      tone=jidint(smc)
c
c      loading the information in the facts table
c
c
c
c      open(unit=9,file='newton.dat',status='new')
c      write(9,900)nodename,ntone
c      write(9,910)nodename,nodename
c 900  format(' smc_tone(m',a<nchars>',' ',i3,')'.)
c 910  format(' smc(' ,a<nchars>',' ',m',a<nchars>',' ').)
c      close (9)
c
c
c      return
c      end
c      subroutine angel(angulo,elevat,a,b,c)
c
c      to obtain the normal direction of a surface
c
c      implicit real*8(a-h,o-z),integer*4(i-n)
c      pi=4.0*datan(one)
c      if(c.gt.0.)go to 90
c 90  if(a.ne.0.0)go to 100
c      angulo=0.0
c      go to 150
c 100  angulo=pi/2.0-datan(dabs(b)/dabs(a))
c 150  if((a.le.0.0).and.(b.ge.0.0))angulo=pi-angulo
c      if((a.le.0.0).and.(b.lt.0.0))angulo=pi+angulo
c      if((a.ge.0.0).and.(b.le.0.0))angulo=2.0*pi-angulo
c      elevat=pi/2.0-dabs(datan(c/(dabs(dcos(angulo)*a))))
c      return
c      end
c      subroutine direct(angulo,elevat,a,b,c)
c
c      to obtain a direction given a single coordinate
c
c      implicit real*8(a-h,o-z),integer*4(i-n)
c      pi=4.0*datan(one)
c      angulo=datan(b/a)+pi
c      elevat=dabs(datan(c/(dabs(dsqrt(a**2+b**2)))))
c      return
c      end

```

## LIST OF REFERENCES

1. Brooks, R.A., Greiner, R., and Binford, T.O., "The ACRONYM Model-Based Vision Systems," *Proc. IJCAI-79*, pp. 103-113, 1979.
2. Rosenthal, D.A., *An Inquiry Driven Computer Vision System Based on Visual and Conceptual Hierarchies*, PhD. Thesis, University of Pennsylvania, 1979.
3. Ballard, D.H., Brown, C.M., and J.A. Feldman, "An approach to Knowledge-Directed Image Analysis," *Proc. IJCAI-77*, pp. 664-870, 1977.
4. Matsuyama, Nagao M., Ikeda, T., and Ikeda, Y., "Region Extraction and Shape Analysis of Aerial Photographs," *Proc. IJCPR-78*, pp. 620-628, 1978.
5. Fischler, M.A., Tenenbaum, J.M., and Wolf, H.C., "Detection of Roads and Linear Structures in Low-Resolution Aerial Imagery Using a Multisource Knowledge Integration Technique," *Computer Graphics and Image Processing 15*, pp. 201-223, 1981.
6. Faugueras, O.D. and Price, K.E., "Semantic descriptions of Aerial Images Using Stochastic Labelling," *Proc. ICJPR-80*, pp. 352-357, 1980.
7. Chien, Yi-Tzuu and Liebowitz, Jay, "Expert Systems in the SDI Environment," *Computer Magazine*, pp. 115-122, July 1986.
8. Mark J. Carlotto, Victor T. Tom, Paul W. Baim, and Richard A. Upton, "Knowledge-Based Multi-spectral image classification," *The Analytic Sciences Corporation*, 1 Jacob Way, Reading, Mass. 01867.
9. Goldberg, M., Karam, G., and Alvo, M., "A Production Rule-Based expert system for interpreting multi-temporal Landsat imagery," *CH1891-1/83/0000/0077*, IEEE, pp. 77-81, 1983.
10. Se June Hong, "Guest Editor's Introduction," *Computer Magazine*, pp. 12-15, July 1986.
11. Nevatia, Ramakant and Price, Keith, "Locating Structures in aerial photographs," *Imaging Processing Institute*, University of Southern California.
12. Selfridge, Peter G. and Sloan, Kenneth R. Jr., "Locating Objects under different conditions: An example of Aerial Image Understanding," *CH1595-8/81/0000/0470*, IEEE, pp. 470-471, 1981.
13. Selfridge, Peter G. and Sloan, Kenneth R. Jr., "Reasoning about success and failure in aerial image understanding," *82CH1761-6/82/0000/0044*, IEEE, pp. 44-49, 1982.

14. Mckeown, David M. Jr. and Pane, John F., "Alignment and Connection of Fragmented Linear Features in Aerial Imagery," *CH2145-1/85/0000/0055*, IEEE, pp. 55-56, 1985.
15. Moffitt, Francis H. and Mikhail, Edward M., *Photogrammetry*, Third Edition, chapters 6 and 9.
16. Zen Chen, Peng-Hu Chiu, and Tsai-Chia Chou "A simplified method for matching sensed planar scenes to reference scenes in a 3-D space," *CH1891-1/83/0000/0366*, IEEE, pp. 366-369, 1983.
17. Robert J. Woodham "A computational vision approach to remote sensing," *CH2145-1/85/0000/0002*, IEEE, pp. 2-11, 1985.
18. Smith, Graham B., "The relationship between image irradiance and surface orientation," *CH1891-1/83/0000/0014*, IEEE, pp. 14-18, 1983.
19. Ballard, D.H. and Sabbah, D., "Detecting Object Orientation from Surface Normals," *CH1595-8/81/0000/0063*, IEEE, pp. 63-67, 1981.
20. Foley, J.D. and Van Dam, A., *Fundamentals of Interactive Computer Graphics*, Addition-Wesley Publishing Company, pp. 457-460, 1982.

# INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22301-6145	2
2. Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3. Professor Chin-Hwa Lee Code 62 Le Computer and Electrical Eng. Dept. Naval Postgraduate School Monterey, California 93943	5
4. Professor Charles W. Therrien. Code 62 Ti Computer and Electrical Eng. Dept. Naval Postgraduate School Monterey, California 93943	1
5. Direccion de Educacion Primera Zona Naval Guayaquil-Ecuador	2
6. Office of the Naval Attache Embassy of Ecuador 2535 15th. St. N.W. Washington D.C. 20009	1
7. LT Diego L. Rueda 650 Sloat Ave. Apt. #1 Monterey, California 93940	1
8. LTJG Carlos Vallejo SMC 2378, Naval Postgraduate School Monterey, California 93943	1









221164

Thesis

R841

Rueda

c.1

Image interpretation  
using an expert system.

221164

Thesis

R841

Rueda

c.1

Image interpretation  
using an expert system.



thesR841

Image interpretation using an expert sys



3 2768 000 76079 7

DUDLEY KNOX LIBRARY